PROTOTIPO WEB PARA LA GENERACIÓN AUTOMÁTICA DE COTIZACIONES SOBRE LOS SERVICIOS CLOUD DE AMAZON WEB SERVICES

JOSDIN DANIEL PINZÓN QUINTERO
CÓD.: 625275

UNIVERSIDAD CATÓLICA DE COLOMBIA
PROGRAMA DE INGENIERÍA DE SISTEMAS
FACULTAD DE INGENIERÍA
BOGOTÁ D.C.
2020

PROTOTIPO WEB PARA LA GENERACIÓN AUTOMÁTICA DE COTIZACIONES SOBRE LOS SERVICIOS CLOUD DE AMAZON WEB SERVICES

JOSDIN DANIEL PINZÓN QUINTERO
CÓD.: 625275

PROYECTO DE GRADO

ASESOR:
PhD. ALEXANDRA MARÍA LÓPEZ SEVILLANO

UNIVERSIDAD CATÓLICA DE COLOMBIA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
BOGOTÁ D.C.
2020

NOTA DE ACEPTACIÓN

Firma del presidente del jurado

Firma del jurado

Firma del jurado

Firma del jurado

Bogotá D.C., 15 de mayo de 2020

DEDICATORIA

A Dios.
Por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos.

A mi Madre,
Por haberme apoyado en todo momento, por sus consejos, sus valores, por la motivación constante y por el valor mostrado para salir adelante.

AGRADECIMIENTOS

En primer lugar, deseo expresar mi agradecimiento a Dios, por brindarme la vida, la salud y la protección que me permite avanzar cada día para lograr mis objetivos; a mi familia, a mi madre, mi hermana y mi esposa, por su paciencia, comprensión y solidaridad con este proyecto, sin su apoyo este trabajo nunca se habría escrito.

De igual forma agradecer a la asesora de tesis de grado, PhD. Alexandra López, por la dedicación, profesionalismo, orientación y apoyo, por brindarme las herramientas y la confianza para el desarrollo de este trabajo.

También quiero agradecer a todos aquellos docentes de la facultad de Ingeniería de la Universidad Católica de Colombia que me apoyaron y me encaminaron a este punto y me brindaron lo mejor de sí mismos, gracias por hacer de sus clases momentos activos de construcción continua de conocimiento.

A todos, muchas gracias.

TABLA DE CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE ANEXOS

RESUMEN

Las empresas tecnológicas que son socios de grandes compañías como Amazon Web Services (AWS) ofrecen servicios de consultoría para la implementación de proyectos de infraestructura nube a diferentes clientes, estos servicios incluyen la estimación del tiempo de ejecución del proyecto y el costo de la infraestructura que se va a utilizar. Diariamente estas empresas consultoras deben considerar los precios de los servicios de AWS y construir una propuesta formal que se ve reflejada como un documento entregable para el cliente.

Para reducir el tiempo de cotización y elaboración de la propuesta, se diseña y se desarrolla un prototipo web que permite estimar los precios de los servicios de infraestructura de AWS, de esta manera las empresas solo tendrán que ingresar los recursos de cómputo que desean cotizar, por ejemplo, servidores virtuales, y la aplicación web se encarga de consultar los precios públicos de AWS por medio de una API, para finalmente generar un documento PDF.

El prototipo es elaborado utilizando la metodología ágil Scrum, ya que esta metodología garantiza flexibilidad y mejora continua durante el desarrollo del software. Adicionalmente la aplicación se implementa utilizando infraestructura nube en Amazon Web Services, cuyo despliegue es automatizado por medio de código utilizando Terraform.

Finalmente, el prototipo se ha presentado en una empresa de consultoría permitiendo abrir nuevas oportunidades de mejora en el desarrollo y evolución de la aplicación.

Palabras clave: Amazon Web Services, AWS, nube, consultoría, infraestructura, cotización, API, Scrum, Terraform, prototipo.

INTRODUCCIÓN

Amazon Web Services (AWS)[1] es una plataforma de nube pública que ofrece un amplio conjunto de servicios de computación como servidores virtuales, bases de datos, distintos tipos de almacenamiento, herramientas para desarrolladores, servicios de aprendizaje automático, servicios de internet de las cosas y muchas más, que se consumen a través de internet, están disponibles en cuestión de segundos y se paga únicamente por lo que se utiliza.

AWS es adecuado para cualquier tipo de empresa a nivel mundial, desde el desarrollador que labora independiente hasta empresas muy grandes como Amazon.com, proporcionando una plataforma flexible sin ninguna inversión inicial donde las compañías puede desplegar cualquier tipo de ambiente permitiéndoles ejecutar sus aplicaciones empresariales, aplicaciones de negocio, aplicaciones móviles y web, etc[1].

Los precios de estos servicios de infraestructura en AWS[2] son como si se tratara de los precios de los servicios de luz o de agua, por ejemplo, si deja la llave abierta del lavamanos el contador empieza a marcar el volumen de agua en metros cúbicos que pasa a través de él y al final del mes llega la factura del agua que se utilizó, de igual forma así funciona el método de pago de estos servicios, sólo paga por lo que se consume.

Con la creciente demanda y aceleración de servicios de cómputo en la nube en Colombia, AWS anuncia la apertura de la primera oficina en Bogotá[1] con la premisa de apoyar a los clientes colombianos en el camino de adopción y transformación de la computación de la nube en el país.

En consecuencia, muchas compañías consultoras han visto la necesidad de familiarizarse y convertirse en socio aliado estratégico de AWS ocasionando una gran competencia en las oportunidades de negocio en el sector de Tecnología de Información (TI). Estos socios aliados también llamados partners se enfocan en el logro de resultados, ayudando a los clientes a aprovechar todos los beneficios que ofrece la nube.

Sin embargo, en el momento de generar una propuesta económica para clientes que están interesados en utilizar los servicios de infraestructura de AWS, este proceso puede llegar a ser muy tedioso hasta el punto de dedicar mucho tiempo en su elaboración, debido a que se debe tener en cuenta que los precios de los servicios de AWS no son fijos y se están actualizando periódicamente.

---

[1] Amazon Web Services, Inc., “Servicios de computación en la nube Colombia | AWS”, 2019. [En línea]. Disponible en: https://aws.amazon.com/es/colombia/

[2] Amazon Web Services, Inc., “Precios de los servicios de la nube | AWS”, 2019. [En línea]. Disponible en: https://aws.amazon.com/es/pricing/

Teniendo en cuenta la necesidad de facilitar la elaboración de propuestas económicas a los partners se propone el desarrollo de un prototipo web llamado *MyCloudPrices* que permita realizar cotizaciones basadas en el precio público de los siguientes servicios de AWS: Amazon Elastic Cloud Computing (EC2), Amazon Simple Storage Services (S3) y Amazon Elastic Block Storage (EBS). *MyCloudPrices* facilitará la tarea de cotizar sistemáticamente el costo del servicio evitando la consulta y calculo manual de precios de los servicios anteriormente mencionados.

Adicionalmente el prototipo web será desplegado bajo una infraestructura automatizada, escalada y altamente disponible en AWS con el fin de demostrar cómo es posible aprovechar los verdaderos beneficios que ofrece la nube pública.

1. PLANTEAMIENTO DEL PROBLEMA

La nube de AWS proporciona un amplio conjunto de servicios de infraestructura[3], con la creciente demanda y aceleración de servicios de cómputo en la nube, surge urgentemente la necesidad de actualizar y renovar en gran parte la infraestructura y aplicaciones existentes on-premises a la nube con el fin de optimizar costos, aumentar la disponibilidad, tener mayor escalabilidad y mejorar el rendimiento.

Sin embargo, hoy en día la problemática de las empresas es conocer el valor exacto que tendrían que invertir si deciden migrar toda o parte de su infraestructura a la nube, por eso, inicialmente estas empresas acuden directamente a los socios certificados por AWS que ofrecen la asesoría y la consultoría profesional para analizar, diseñar, proponer, e implementar todo un proceso de migración y mejores prácticas. No obstante, estos partners encuentran dificultad a la hora de plantear y exponer propuestas económicas ya que no existe una herramienta integral que permita calcular de forma precisa el valor individual y total de cada servicio.

Como un primer acercamiento a la solución del problema se propone el diseño y desarrollo de un prototipo web llamado *MyCloudPrices* para la generación de cotizaciones de los servicios web brindados por AWS.

Las preguntas que se tratarán de resolver son:

- ¿Cómo reducir el tiempo de estimación y elaboración de propuestas económicas manualmente?

- ¿Cómo capturar la información de los costos obtenidos desde la página web oficial de Amazon Web Services?

- ¿Cómo garantizar disponibilidad y escalabilidad del servicio ofrecido?

[3] Amazon Web Services, Inc., “What is Cloud Computing? - Amazon Web Services”, 2019. [En línea]. Disponible en: https://aws.amazon.com/es/what-is-aws/

2. OBJETIVOS

2.1. OBJETIVO GENERAL

Desarrollar un prototipo web altamente escalable que permita elaborar cotizaciones con los precios públicos de los servicios que ofrece Amazon Web Services utilizando tecnologías de desarrollo en la nube.

2.2. OBJETIVOS ESPECÍFICOS

- Identificar los requerimientos funcionales y no funcionales del sistema para determinar las necesidades del servicio.

- Diseñar la arquitectura nube con el propósito de desplegar la aplicación web de manera automatizada.

- Construir un método automático para la captura de precios actuales de los servicios que se encuentran publicados en AWS.

- Implementar un prototipo funcional que permita gestionar cotizaciones de infraestructura nube AWS.

3. JUSTIFICACIÓN

Debido a la competencia actual en el mundo de la tecnología, las pequeñas, medianas y grandes empresas se han encontrado con diversos obstáculos y barreras tecnológicas tales como: La utilización de infraestructura en sitio propia o tercerizada que se devalúa y se desactualiza rápidamente con el tiempo, la alta demanda en el mercado por consumir servicios mucho más económicos y ágiles en la nube y, la exigencia operacional para dejar a un lado los servicios o productos on-premises que ocasionan un mayor gasto en el mantenimiento y operación. Es por esto que surge la necesidad de migrar y utilizar los nuevos servicios de infraestructura en la nube de Amazon Web Services.

La tecnología está en constante evolución y las empresas deben reinventarse para sobrevivir a los nuevos ecosistemas digitales del siglo XXI, de modo que, las empresas deben desarrollar una estrategia de transformación digital para asegurar su supervivencia. Los nuevos hábitos de los clientes, la aparición de modelos de negocio distintos propiciados por lo digital y la amenaza de la irrupción de competidores digitales obliga a las compañías tradicionales a innovar y ser más eficientes.

El desconocimiento de este cambio tecnológico produce que empresas consultoras o partners de dichas tecnologías apoyen a estas compañías de una manera estratégica a esta transformación digital haciendo uso en parte, del sistema *MyCloudPrices* para crear cotizaciones que permita a los clientes conocer el costo que implica llevar a su compañía a otro nivel más competitivo.

Finalmente, con *MyCloudPrices* se busca generar y brindar de una manera fácil y confiable una cotización que los partners puedan utilizar para vender a sus clientes los servicios de infraestructura AWS y así cerrar negocios más rápidamente.

4. ALCANCE Y LIMITACIONES

4.1. ESPACIO

El proyecto se llevará a cabo en gran parte en el lugar de residencia del autor y en algunas ocasiones en las instalaciones de la Universidad Católica utilizando infraestructura de AWS en una cuenta personal para el despliegue y pruebas del prototipo web.

4.2. TIEMPO

El desarrollo del proyecto está pensado para ser desarrollado en 6 meses hasta la socialización de trabajos de grados a finales del primer semestre del 2020.

4.3. CONTENIDO

El proyecto contiene toda la documentación necesaria de la planeación, diseño, desarrollo, código fuente, pruebas y despliegue piloto del prototipo *MyCloudPrices* sobre la plataforma de AWS.

4.4. ALCANCE

El prototipo web tendrá en cuenta únicamente los precios de los servicios de Amazon EC2, Amazon S3 y Amazon EBS en la región de Virginia, ya que son los más comúnmente utilizados en las implementaciones nube de AWS en Colombia.

Como prueba piloto, el sistema de información quedará alojado en la nube sobre infraestructura AWS arrendada por el autor durante el tiempo de la entrega del trabajo, finalmente se entregará la documentación y código fuente a la Universidad Católica de Colombia.

4.5. LIMITACIONES

No se incluye el resto de los servicios que ofrece AWS aparte de los mencionados en el alcance de este proyecto. Posterior a la fecha se tiene planeado continuar con el proyecto de forma independiente añadiendo nuevas funcionalidades y servicios.

5. ANTECEDENTES

Los precios de los servicios nube de AWS están basados (dependiendo del servicio) en los siguientes factores: tiempo por hora consumido, espacio de almacenamiento aprovisionado o usado en gigas (GB), cantidad de recursos y forma de pago.

Inicialmente para estimar cuanto podría costar un servicio en AWS se tenía que calcular manualmente con operaciones aritméticas el precio del día, semana o mes según se requería, esto era demasiado tedioso y generaba cierta inseguridad a los usuarios por temor a no realizar los cálculos correctamente, sin embargo, para ofrecer una mayor confiabilidad AWS elaboró una calculadora web que permite estimar el precio mensual de algunos (no todos) de sus servicios, ver la Figura 1. Esta calculadora se encuentra disponible en la web[4].

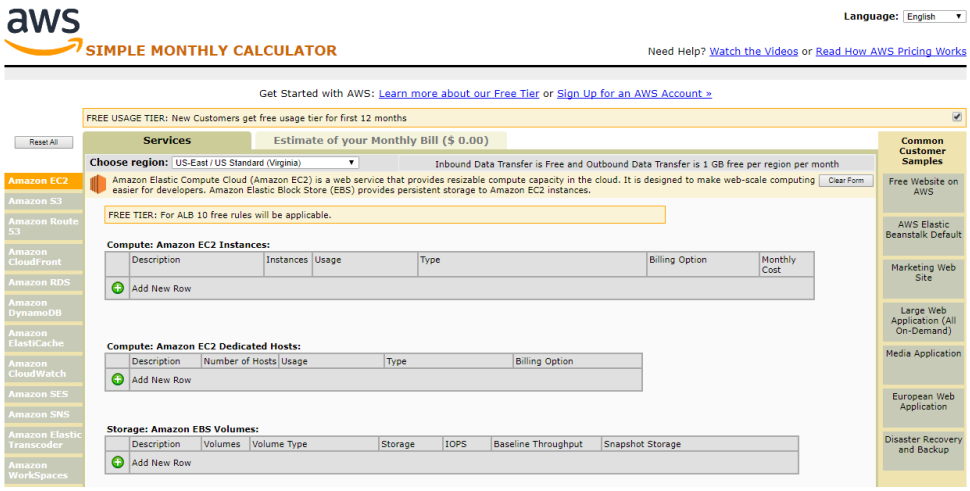


Figura 1. Calculadora pública de AWS

A pesar de que esta calculadora web generó cierta confianza a las empresas, hoy en día los precios de los servicios de AWS no son fijos y pueden cambiar, por lo general bajan de precio, y esta reducción no se ve reflejada de forma inmediata en la calculadora, por lo que es necesario esperar entre 3 a 5 días para utilizar el precio actualizado. Esto es importante ya que un precio de un servicio que ha sido reducido puede influenciar en la toma de decisión de las compañías de utilizar servicios de infraestructura en la nube o continuar con el esquema tradicional de centros de datos físicos.

[4] Amazon Web Services, Inc., “Simple Monthly Calculator”, 2019. [En línea]. Disponible en: https://calculator.s3.amazonaws.com/index.html

Por otra parte, la exportación de la información de la calculadora no es muy amigable. En este ejemplo se está cotizando dos (2) servidores web con sistema operativo Linux, un almacenamiento de 8GB para cada uno, un servicio de DNS (Domain Name System) y una base de datos relacional PostgreSQL, Ver Figura 2.

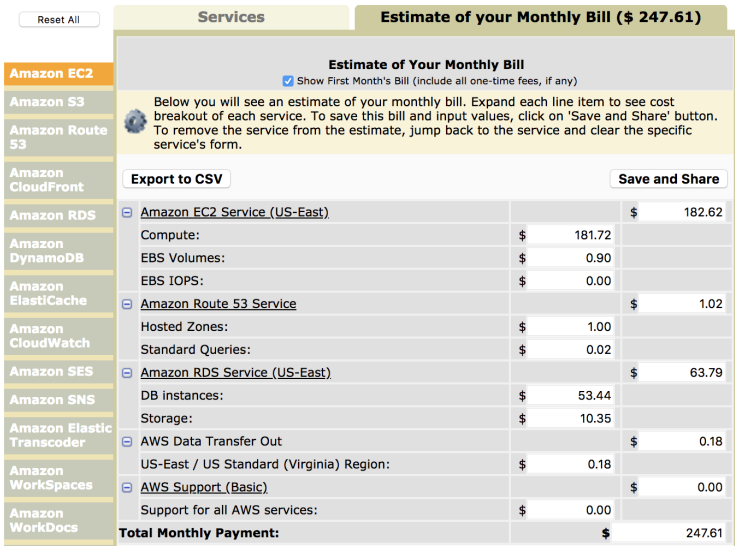


Figura 2. Ejemplo Cotización AWS

La calculadora permite exportar la estimación a formato Excel como se muestra en la Figura 3.

	A	B	C	D	E
1	Your Estimate				
2	Service Type	Components	Region	Component F	Service Price
3	Amazon EC2 Service (US-East)				$182.62
4		Compute:	US-East / US	$181.72	
5		EBS Volumes	US-East / US	$0.9	
6		EBS IOPS:	US-East / US	$0	
7	Amazon Route 53 Service				$1.02
8		Hosted Zone	Global	$1	
9		Standard Qu	Global	$0.02	
10	Amazon RDS Service (US-East)				$63.79
11		DB instances	US-East / US	$53.44	
12		Storage:	US-East / US	$10.35	
13	AWS Data Transfer Out				$0.18
14		US-East / US	Global	$0.18	
15	AWS Support (Basic)				$0
16		Support for all AWS servic		$0	
17			Total Monthly Payment:		$247.61

Figura 3. Exportación precios Calculadora AWS

Como se puede observar no es muy intuitivo visualizar el precio de los servicios y tampoco es posible discriminar individualmente el precio de cada servidor con su almacenamiento respectivo, algo que solicitan comúnmente la mayoría de los clientes.

Por otro lado no todos los servicios que ofrece AWS se encuentran disponibles para calcular su costo mensual, por ejemplo: Amazon DMS (Database Migration Service), AWS CodeCommit, AWS CodeStart, AWS CodeBuild, AWS CodeDeploy, AWS WAF (Web Application Firewall), Amazon Lex, Amazon Polly, Amazon Machine Learning, etc., estos son servicios que aún no cuentan con una sección que facilite la estimación de consumo y hace que sea obligatorio calcular manualmente el precio.

Otro claro ejemplo es con el servicio de Amazon WorkDocs, este es un servicio de administración, almacenamiento e intercambio de archivos seguro (muy parecido al servicio de Google Drive o Microsoft OneDrive), lo usuarios pueden compartir archivos, brindar comentarios y obtener acceso a sus archivos almacenados desde cualquier dispositivo, este servicio tiene un costo de $5 dólares por usuario con un almacenamiento inicial de 200GB, AWS extendió su servicio ofreciendo 1TB de almacenamiento por usuario **sin coste adicional**, sin embargo, en la calculadora web de AWS esta actualización no se ve reflejada por tanto la estimación de precio no es correcta. Ver Figura 4.

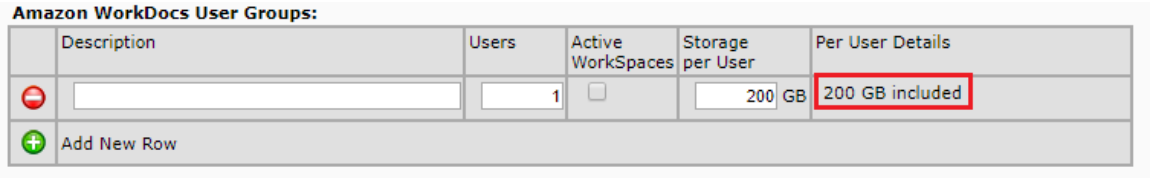


Figura 4. Precio de Amazon WorkDocs en la calculadora pública de AWS

Asimismo, todos los años AWS realiza un evento llamado “AWS re:Invent” en el cual anuncia los nuevos productos y servicios que van a permitir a los usuarios mejorar y agilizar sus tareas de TI haciendo uso de las últimas tecnologías disponibles desde cualquier parte del mundo. A final del año 2017 AWS anunció un nuevo tipo de servidores que se encontraban disponibles al público para su uso, al intentar calcular el precio de estos nuevos servidores en la calculadora de AWS, aún no se encontraban en el listado de servidores, pero si en la página web, por tanto en su momento, fue necesario hacer el cálculo manual para tener una estimación aproximada del costo mensual; pasados unos meses el precio de estos nuevos servidores ya se encontraban disponibles en la calculadora de AWS.

Finalmente, hoy en día las empresas partners utilizan plantillas en Excel con macros personalizados para estimar los costos de los servicios en la nube de AWS, esto puede ser una solución rápida para cotizar los servicios que serán ofrecidos a los clientes, sin embargo, si ocurre una actualización de precios, estas empresas se ven obligadas a generar manualmente una nueva versión del archivo Excel e informar a cada uno de sus colaboradores y sucursales del nuevo cambio. Y, hasta el momento por conocimiento del autor, no se ha encontrado aplicaciones

o sistemas similares que permitan cotizar de una manera más confiable y automatizada.

6. MARCO DE REFERENCIA

6.1. MARCO TEÓRICO

El proyecto contiene un conjunto de recomendaciones y buenas prácticas para el desarrollo del sistema de información. En este caso se utiliza la metodología Ágil SCRUM combinada con DevOps, las cuales permiten el inicio, la planeación, el desarrollo y el despliegue del sistema de información de una manera mucho más ágil.

6.1.1. ¿QUE ES SCRUM?

Un proyecto de Scrum implica un esfuerzo colaborativo para crear un producto o servicio nuevo, o cualquier otro resultado según lo que defina la declaración de visión del proyecto. Los proyectos se ven afectados por limitaciones de tiempo, costo, alcance, calidad, recursos, capacidades organizacionales y demás limitaciones que dificultan su planificación, ejecución, gestión y por último su éxito. Sin embargo, la implementación exitosa de los resultados de un proyecto terminado proporciona considerables beneficios de negocio a una organización. Por lo tanto, es importante que las organizaciones seleccionen e implementen un apropiado método para la gestión de proyectos[5].

Scrum es uno de los métodos ágiles más populares. Es un framework adaptable, iterativo, rápido, flexible y eficaz, diseñado para ofrecer un valor considerable en forma rápida a lo largo del proyecto. Scrum garantiza transparencia en la comunicación y cera un ambiente de responsabilidad colectiva y de progreso continuo. El framework de Scrum está estructurado de tal manera que es compatible con el desarrollo de productos y servicios en todo tipo de industrias y en cualquier tipo de proyecto, independientemente de su complejidad[5].

Un principio clave de Scrum es el reconocimiento de que durante un proyecto los clientes pueden cambiar de idea sobre lo que quieren y necesitan, y que los desafíos impredecibles no pueden ser fácilmente enfrentados de una forma predictiva y planificada. Por lo tanto, Scrum adopta una aproximación pragmática, aceptando que el problema no puede ser completamente entendido o definido, y centrándose en maximizar la capacidad del equipo de entregar rápidamente y responder a requisitos emergentes[6].

---

[5] Scrumstudy.com, “Una guía para el CUERPO DE CONOCIMIENTO DE SCRUM | GUÍA SBOK – Tercera Edición”, 2019. [En línea]. Disponible en: https://www.scrumstudy.com/SBOK/SCRUMstudy-SBOK-Guide-3rd-edition-spanish.pdf

[6] V. C. J. Demetrio, “Desarrollo en Cascada (Waterfall) VS Desarrollo Agile-SCRUM,” 2013. [En línea]. Disponible en: http://www.northware.mx/wp-content/uploads/2013/04/Desarrollo-cascada-vs-Desarrollo-Agile.pdf.

Existen varias implementaciones de sistemas para gestionar el proceso de Scrum, que van desde notas amarillas "post-it" y pizarras hasta paquetes de software. Una de las mayores ventajas de Scrum es que es muy fácil de aprender, y requiere muy poco esfuerzo para comenzarse a utilizar[6].

¿POR QUÉ SCRUM?

MÉTODOS TRADICIONALES

Unos de los métodos tradicionales de desarrollo de software más utilizados es el llamado modelo “cascada”, que consiste en que un software se desarrolla en etapas, debiéndose concluir cada una de las anteriores para comenzar la siguiente[7]. Ver Figura 5.

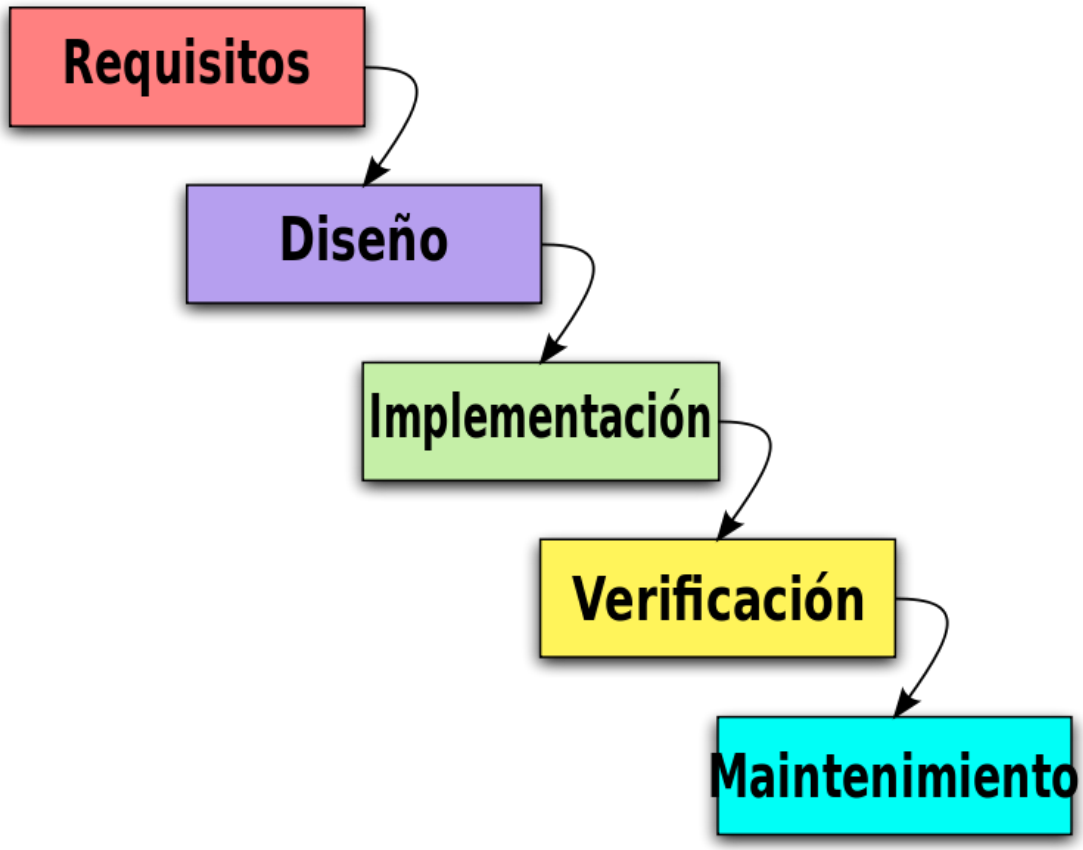


Figura 5. Modelo Cascada.
Tomado de https://es.wikipedia.org/wiki/Desarrollo_en_cascada

[7] SCRUM TECHNOTIVIT. “Curso de Scrum” 2018. [En línea]. Disponible en la intranet de http://intranet-latam.tivit.com/.

Si se utilizan métodos de desarrollo tradicionales, cualquier modificación exigida tras el análisis provoca un “retorno” a la fase inicial. Además, si se entiende mal algún requisito o este no se ajusta a lo esperado, el usuario final solamente lo notará después de la implementación. Una característica de las metodologías tradicionales, como la cascada, es que en cada fase pueden actuar equipos diferentes, lo que puede causar ruidos en la comunicación y falta de integración entre los profesionales[7].

MÉTODOS ÁGIL

Para minimizar algunos de estos problemas y asegurar que se desarrollen software de calidad, que atiendan las expectativas del cliente y también asegure la optimización de los plazos y costos de los proyectos, los especialistas del desarrollo crearon formas de trabajo perfeccionadas, entre ellas, la llamada Metodología de Desarrollo Ágil[7].

La filosofía de los métodos agiles se fundamenta sólidamente en la integración de los equipos, de modo que se asegura la productividad, calidad y flexibilidad del trabajo de desarrollo. El desarrollo ágil aspira a lograr un proceso que fomente la verificación y adaptación frecuentes, y una filosofía que fomente el trabajo en equipo[7].

Los métodos agiles integran los equipos del proyecto de desarrollo y acercan al cliente el proceso, en la medida en que, en vez de aguardar la conclusión del desarrollo, el cliente comienza a validar partes funcionales del software y puede contribuir al proyecto con sugerencias y eventuales correcciones y ajustes. Este proceso se denomina desarrollo incremental[7]. Ver Figura 6.

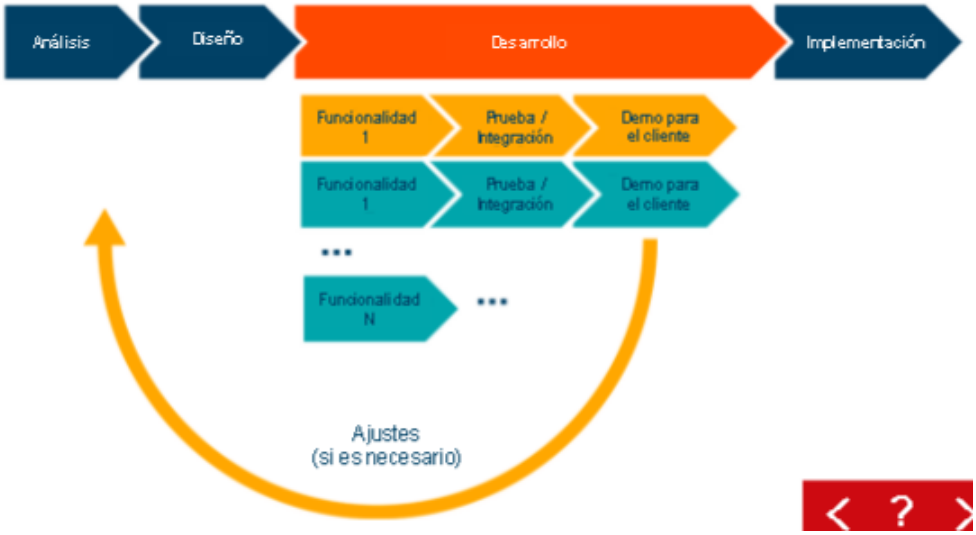


Figura 6. Método ágil.
Tomado de http://intranet-latam.tivit.com/

La aplicación de metodologías de desarrollo ágil da lugar a una mayor interacción entre los equipos de análisis, desarrollo y pruebas, además de presentar resultados mucho más rápidos al cliente y a los demás stakeholders del proyecto. Los eventuales ajustes se corrigen en las fases intermedias, optimizando el tiempo de desarrollo y mejorando los plazos y costos del proyecto[7].

A continuación, se hace una comparación entre Scrum vs la gestión de proyectos tradicionales con el fin de mostrar las ventajas de utilizar Scrum en este proyecto.

	SCRUM	**TRADICIONAL**
Énfasis	En las personas	En los procesos
Documentación	Mínima; solo lo necesario	Excesiva
Estilo de procesos	Iterativo	Lineal
Planificación por adelantado	Baja	Alta
Priorización de requerimientos	Basada en el valor del negocio y se actualiza constantemente	Fija en el plan del proyecto
Garantía de calidad	Centrada en el cliente	Centrada en procesos
Organización	Auto-organizada	Gestionada
Estilo de gestión	Descentralizado	Autocrático
Cambio	Actualizaciones al Backlog Priorizado del producto	Estilo formal de gestión de cambios
Liderazgo	Colaborativo; liderazgo servicial	Comando y control
Medida del desempeño	Valor del negocio	Conformidad con el plan
Retorno sobre la inversión	Temprano / a lo largo del proyecto	A final de la vida del proyecto
Participación del cambio	Alta a lo largo del proyecto	Varía dependiendo del ciclo de vida del proyecto

Tabla 1. Scrum vs Gestión tradicional de proyectos.
Tomado de https://www.scrumstudy.com

El Scrum framework está guiado por el objetivo de ofrecer el máximo valor empresarial en un mínimo período de tiempo. Para lograr esto en forma práctica, Scrum cree en el desarrollo iterativo de entregables. El beneficio del desarrollo iterativo permite la corrección a medida que todas las personas involucradas obtengan una mejor comprensión de lo que se debe entregar como parte del proyecto, e incorporar lo aprendido de una manera iterativa[5].

Como se ilustra en la Figura 7, los proyectos Scrum se completan de manera iterativa, entregando valor a lo largo del clico de vida del proyecto[5].

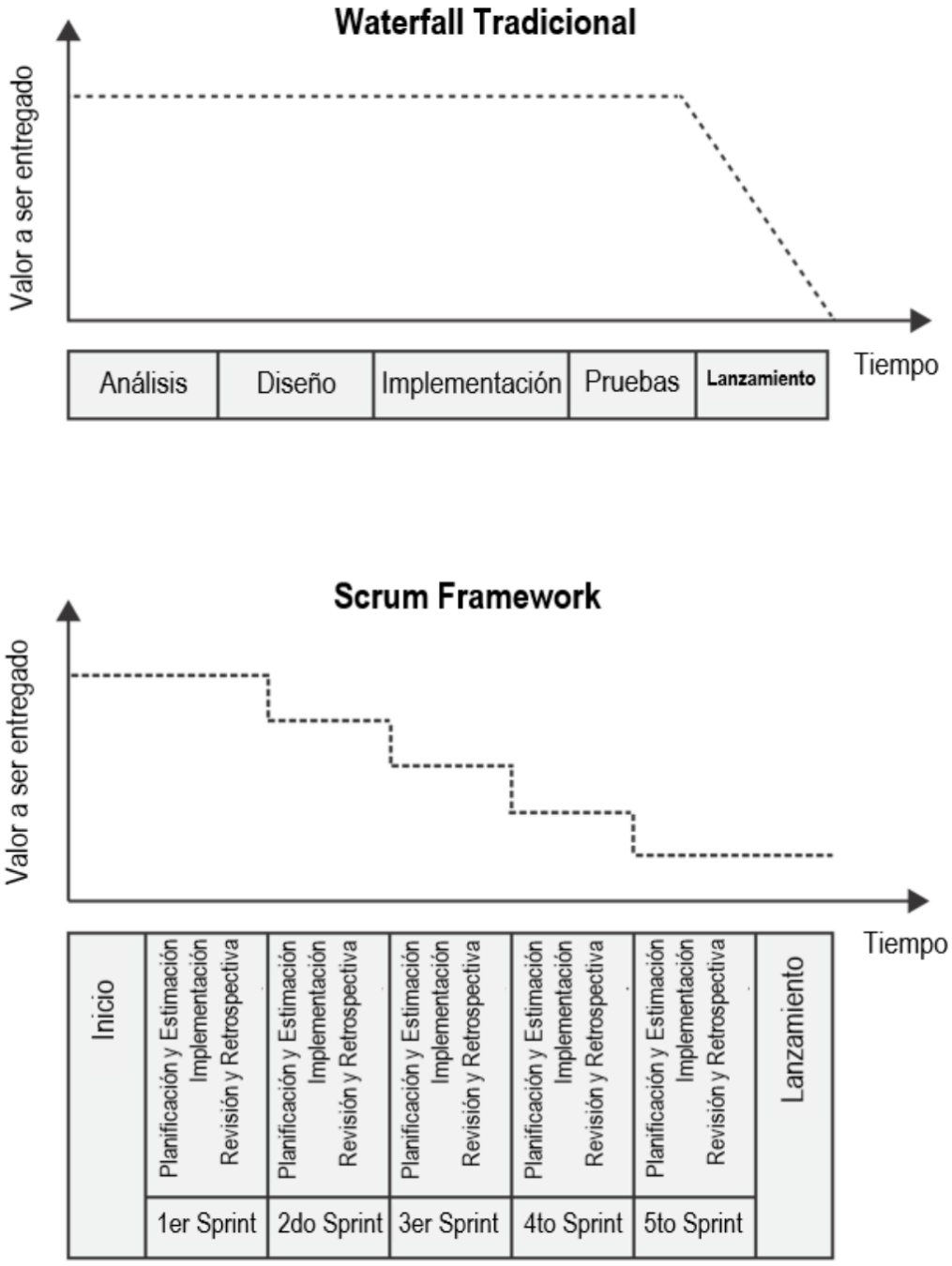


Figura 7. Scrum vs Cascada tradicional
Tomado https://www.scrumstudy.com

Eda

BENEFICIOS DE SCRUM

Algunas de las ventajas principales del uso de Scrum en cualquier proyecto son:

- **Adaptabilidad:** El control de proceso empírico y la entrega iterativa hacen que los proyectos sean adaptables y abiertos a la incorporación del cambio.

- **Transparencia:** Todos los emisores de información son compartidos, lo cual lleva a un ambiente laboral abierto.

- **Retroalimentación constante:** Los procesos de reunión diaria y revisión del sprint permiten una constante retroalimentación.

- **Mejora continua:** Los entregables mejoran progresivamente sprint tras sprint mediante el proceso de refinar el backlog priorizado del producto.

- **Entrega constante de valor:** El proceso iterativo permite la entrega constante de valor, tan frecuentemente como el cliente lo requiera.

- **Ritmo sostenido:** Los procesos de Scrum están diseñados para que las personas involucradas puedan trabajar a un ritmo sostenido al que pueden, en teoría, continuar de forma indefinida.

- **Entrega anticipada de alto valor:** El proceso de Crear el backlog priorizado del producto garantiza que los requerimientos de alto valor del cliente se cumplan primero.

- **Desarrollo de proceso eficiente:** La asignación de un tiempo específico (Time-boxing) y la reducción del trabajo no esencial llevan a niveles más altos de eficiencia.

- **Motivación:** Los procesos de realizar la reunión diaria y retrospectiva del sprint conducen a mayores niveles de motivación entre los empleados.

- **Resolución de problemas en forma más rápida**: La colaboración y la co-ubicación de equipos interfuncionales conducen a la resolución de problemas con mayor rapidez.

- **Entregables efectivos:** El proceso de crear el backlog priorizado del producto y las revisiones periódicas después de la creación de entregables aseguran entregas eficientes al cliente.

- **Centrado en el cliente:** El énfasis en el valor del negocio y contar con un enfoque colaborativo con los stakeholders asegura un framework orientado al cliente.

- **Ambiente de alta confianza:** Scrum promueven la transparencia y la colaboración, llevando a un ambiente laboral de alta confianza y garantizando una baja fricción entre los empleados.
- **Alta velocidad:** Un framework colaborativo permite que los equipos interfuncionales altamente cualificados libren todo su potencial y una alta velocidad.

- **Ambiente innovador:** Scrum crea un ambiente de introspección, aprendizaje y adaptabilidad, lo cual resulta en un ambiente laboral innovador y creativo[5].

PRINCIPIOS DE SCRUM

Los principios de Scrum son las principales pautas para la aplicación del framework de Scrum y deben implementarse obligatoriamente en todos los proyectos Scrum:

- **Control de proceso empírico** – Este principio enfatiza la filosofía central de Scrum con base a las tres ideas principales de transparencia, inspección y adaptación.

- **Auto-organización** – Este principio se enfoca en los trabajadores de hoy en día, que entregan un valor considerable mayor cuando se auto-organizan, lo cual resulta en equipos que poseen un gran sentido de compromiso y responsabilidad; a su vez, esto produce un ambiente innovador y creativo que es más propicio para el crecimiento.

- **Colaboración** – Este principio se centra en las tres dimensiones básicas relacionadas con el trabajo colaborativo: conocimiento, articulación y apropiación. También fomenta la gestión de proyectos como un proceso de creación de valor compartido con equipos que trabajan e interactúan conjuntamente para ofrecer el mayor valor.

- **Priorización basada en valor** – Este principio pone de relieve el enfoque de Scrum para ofrecer el máximo valor de negocio, desde el principio del proyecto hasta su conclusión.

- **Time-Boxing** – Este principio describe cómo el tiempo se considera una restricción limitante en Scrum, y cómo este se utiliza para ayudar a manejar eficazmente la planificación y ejecución del proyecto. Los elementos del time boxing en Scrum incluye sprints, reuniones diarias, reuniones de planificación del sprint y reuniones de revisión del spring.

- **Desarrollo iterativo** – Este principio define el desarrollo iterativo y hace énfasis en cómo gestionar mejor los cambios y crear productos que satisfagan las necesidades del cliente. También delinea las responsabilidades del Product Owner y las de la organización relacionadas con el desarrollo iterativo[5].

La Figura 8 ilustra los seis principios de Scrum.

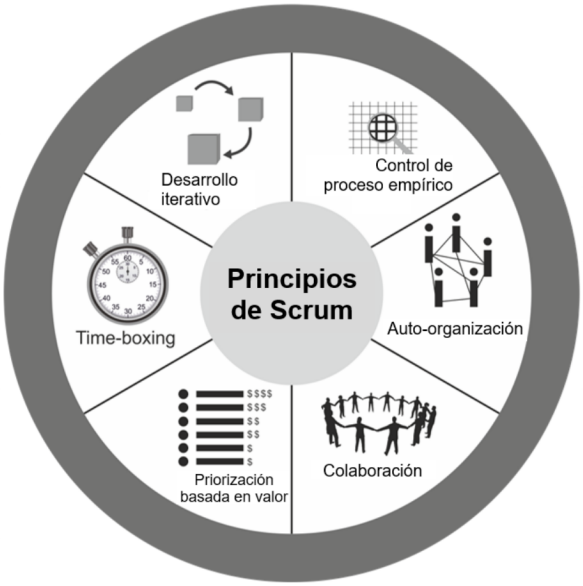


Figura 8. Principios de Scrum
Tomado de https://www.scrumstudy.com.

ORGANIZACIÓN DE SCRUM

Entender los roles y responsabilidades definidos en un proyecto scrum es muy importante a fin de asegurar la implementación exitosa de Scrum.

Los roles de Scrum incluyen:

- El **Product Owner** es la persona responsable de lograr el máximo valor empresarial para el proyecto. Este rol también es responsable de la articulación de requisitos del cliente y de mantener la justificación del negocio para el proyecto. El Product Owner representa la voz del cliente.

- El **Scrum Master** es un facilitador que asegura que el Equipo Scrum cuente con un ambiente propicio para completar el proyecto con éxito. El Scrum Master guía, facilita y enseña las prácticas de Scrum a todos los

involucrados en el proyecto; elimina los impedimentos que pueda tener el equipo y se asegura de que se estén siguiendo los procesos de Scrum.

- El **Equipo Scrum** es el grupo o equipo de personas responsables de entender los requisitos especificados por el Producto Owner y de crear los entregables del proyecto[5].

La Figura 9 ilustra la estructura de la organización Scrum

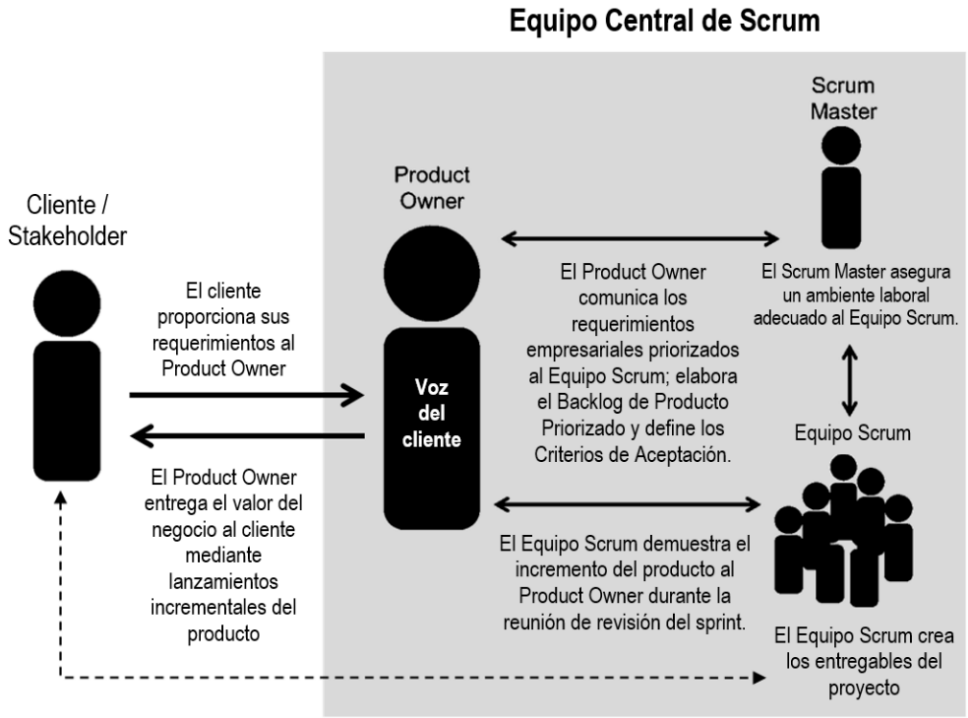


Figura 9. Organización en Scrum
Tomado de https://www.scrumstudy.com

FLUJO DE SCRUM

Una fortaleza clave de Scrum radica en el uso de equipos interfuncionales, auto-organizados y empoderados que dividen su trabajo en ciclos de trabajo cortos y concentrados llamados Sprints. La Figura 10 proporciona una visión general del flujo de un proyecto Scrum[5].

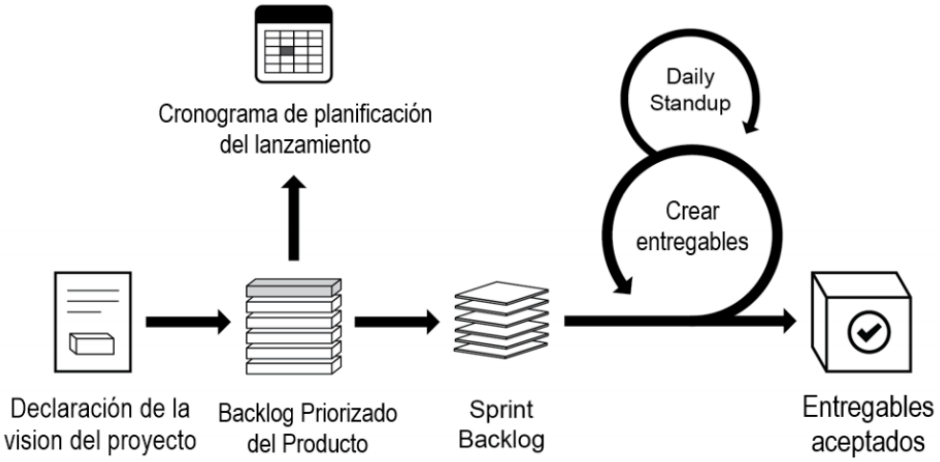


Figura 10. Flujo de Scrum para un sprint
Tomado https://www.scrumstudy.com

El ciclo de Scrum empieza con una reunión de stakeholders, durante la cual se crea la visión del proyecto. Después, el Product Owner desarrolla una Backlog Priorizado del Producto que contiene una lista requerimientos del negocio y del proyecto por orden de importancia en forma de una historia de usuario. Cada sprint empieza con una reunión de planificación del sprint durante la cual se consideran las historias de usuario de alta prioridad para su inclusión en el sprint. Un sprint generalmente tiene una duración de una a seis semanas durante las cuales el Equipo Scrum trabaja en la creación de entregables en incrementos del producto. Durante el sprint, se llevan cabo las reuniones diarias muy breves y concretos, donde los miembros del equipo discuten el progreso diario. Hacia el final del sprint, se lleva a cabo una Reunión de Revisión del Sprint en la cual se proporciona una demostración de los entregables al Product Owner y a los stakeholders relevantes. El Product Owner acepta los entregables solo si cumplen con los criterios de aceptación predefinidos. El ciclo del sprint termina con una Reunión de Retrospectiva del Sprint, donde el equipo analiza las formas de mejorar los procesos y el rendimiento a medida que avanzan al siguiente sprint[5].

6.1.1.1. ¿QUE ES DEVOPS?

DevOps es un nuevo término que se centra principalmente en una colaboración, comunicación e integración mejoradas entre los desarrolladores de software y las operaciones de TI. Es un término general que algunos describen como filosofía, cambio cultural y cambio de paradigma[8].

[8] Amazon Web Services, Inc., “Introduction to DevOps on AWS”, 2014 [En línea]. Disponible en: https://d1.awsstatic.com/whitepapers/AWS_DevOps.pdf

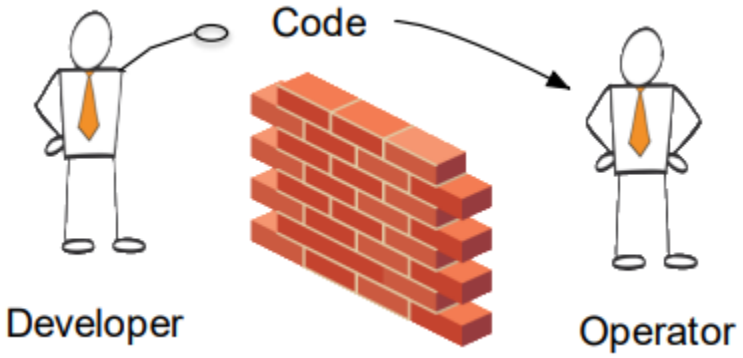


Figura 11. Introducción a DevOps
Tomado de https://aws.amazon.com/es/whitepapers/

Los roles del personal del departamento de TI y desarrollador se fusionan y siguen una serie de principios sistemáticos:

- Infraestructura como código
- Despliegue continuo
- Automatización
- Monitoreo
- Seguridad[8].

En otras palabras, DevOps puede entenderse como un conjunto de prácticas y valores que, de forma comprobada, ayudan a las organizaciones a mejorar sus ciclos de producción de software, su calidad, seguridad y velocidad de desarrollo[9].

El nombre “DevOps” se creó por medio de la unión de las palabras “Development” y “Operations”, y ello ayuda a explicar su concepto, que es el de fomentar la colaboración y la integración entre los desarrolladores de software y los profesionales de operación de los sistemas (profesionales de TI) [8].

En los últimos años, la evolución ágil del desarrollo de software ha comenzado a descender hacia la infraestructura bajo la etiqueta DevOps. Mientras que el desarrollo de software ágil se centra principalmente en la colaboración entre la empresa y sus desarrolladores, DevOps se centra en la colaboración entre desarrolladores, operaciones de TI y equipos de seguridad. Las operaciones de TI incluyen administradores de sistemas, administradores de base de datos, ingenieros de red, arquitectos de infraestructura y personal de soporte. Mientras que el desarrollo de software ágil proporciona agilidad empresarial, DevOps proporciona agilidad de TI, lo que permite la implementación de aplicaciones que son más confiables, predecibles y eficientes[9]. Ver Figura 12.

[9] DEVOPS TECHNOTIVIT. “Curso de Devops” 2018. [En línea]. Disponible en la intranet de http://intranet-latam.tivit.com/

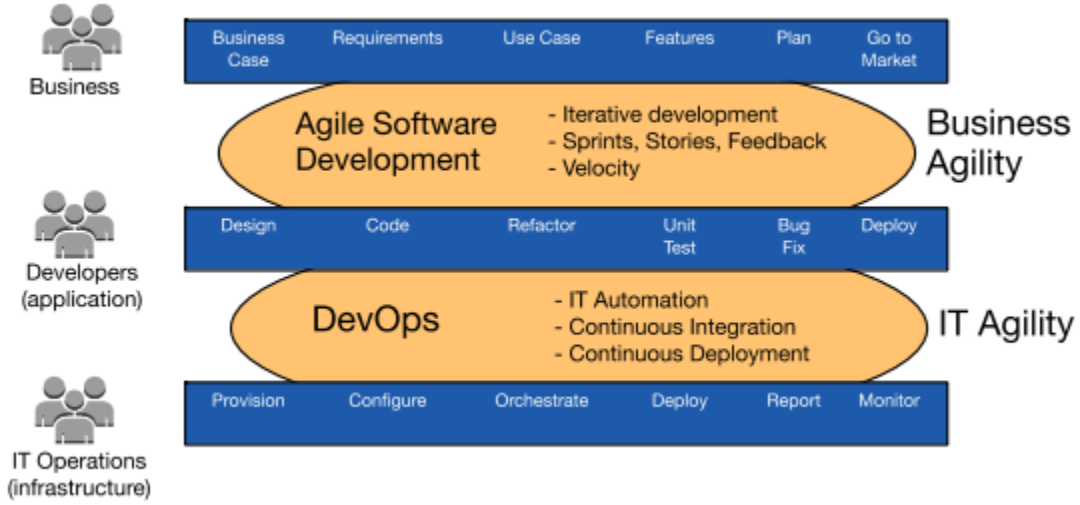


Figura 12. La coevolución del desarrollo de software ágil y DevOps
Tomado de https://aws.amazon.com/es/whitepapers/

OBJETIVO DE DEVOPS

El objetivo de la integración de los equipos de desarrollo y de operación es mejorar la productividad del ciclo de desarrollo, por medio de la automatización de los flujos de pruebas, correcciones y mediciones de desempeño, desarrollando el software en las mejores condiciones, de acuerdo con su finalidad de uso[9].

PRINCIPIOS DE DEVOPS

En el libro The Phoenix Project, los autores describen tres principios básicos de DevOps

- **System Thinking o pensar en el sistema como un todo:** La idea es pensar siempre en el producto final y en el que se entregará al cliente y agregar valor a los negocios, independientemente de las etapas intermedias.

- **Amplify Feedback Loops o crear loops ampliados:** El objetivo es crear un ciclo continuo de mejora del proceso, a fin de incorporar a este proceso las eventuales correcciones y ajustes.

- **Culture of continual experimentation and learnig o crear una cultura de experimentación y aprendizaje continuos:** Este principio defiende la

repetición y la practica como forma de perfeccionamiento y aprendizaje, ya sea con respeto a aquello que salió bien o presentó errores[9].

BENEFICIOS DE DEVOPS

- Más enfoque en lo que impacta en los negocios.
- Participación continua de los clientes.
- Rapidez en la entrega del software.
- Capacidad de innovar más rápidamente.
- Mejor calidad del producto final.
- Mayor colaboración entre los desarrolladores.
- Más frecuencia en la liberación de releases[9].

6.1.1.2. LENGUAJE DE PROGRAMACIÓN Y FRAMEWORK

El lenguaje de programación en el que se desarrollará el sistema de información es JavaScript, utilizando el framework MEAN Stack, el cual, lo conforman 4 tecnologías esenciales: **M**ongoDB, **E**xpress, **A**ngularJS y **N**ode.Js. Este conjunto de tecnologías relacionadas ofrece todo lo necesario para realizar una aplicación web moderna, escalable, testable y mantenible.

Una de las principales ventajas de MEAN es que emplea el mismo lenguaje de programación en todas las partes de la aplicación lo que permite que una persona pueda manejarse en todos los ámbitos de una aplicación web moderna, aunque se especialice en uno de ellos. De esta manera se colabora más en los proyectos y el desarrollo es más continuo[10]. Ver Figura 13.

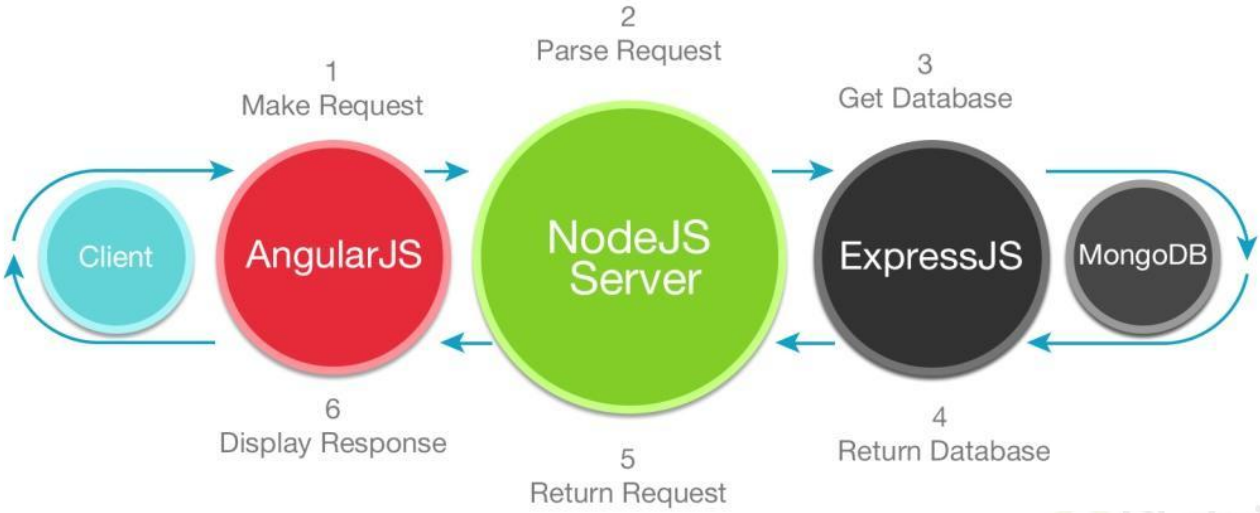


Figura 13. Componentes MEAN Stack.
Tomado de https://platzi.com/blog/que-es-mean-full-stack-javascript/.

A continuación, se explica cada una de ellas:

Mongo, es una base de datos no relacional (NoSQL) de código abierto que guarda los datos en documentos tipo JSON (JavaScript Object Notation) pero en forma binaria (BSON) para hacer la integración de una manera más rápida. Se pueden ejecutar operaciones en JavaScript en su consola en lugar de consultas SQL. Además, tiene una gran integración con Node.js. Debido a su flexibilidad es muy escalable y ayuda al desarrollo ágil de proyectos web[10].

Express, es un framework o marco de trabajo desarrollado sobre Node.js, lo que quiere decir que provee toda la funcionalidad que se necesita para hacer aplicaciones web, permitiendo estructurar un proyecto de una manera más fácil. Express.js es considerado un framework minimalista, es decir que viene con funcionalidades muy básicas, que luego a medida que crezca la aplicación se puede ir extendiendo a través de los módulos que puedes encontrar en npm[11].

AngularJS, es un framework JavaScript para la parte cliente o Frontend de una aplicación web, que respeta el paradigma MVC y permite crear Single-Page Applications (Aplicaciones web que no necesitan recargar la página), de manera más o menos sencilla. Es un proyecto mantenido por Google[10].

Node.js, es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) para el Backend basado en el motor V8 de JavaScript del navegador Google Chrome y orientado a eventos, no bloqueante, lo que lo hace muy rápido a la hora de crear servidores web[10].

6.1.1.3. INFRAESTRUCTURA Y AUTOMATIZACIÓN

La infraestructura que se va a utilizar para implementar el prototipo web *MyCloudPrices* será diseñada y desplegada sobre la plataforma nube AWS, la implementación de esta infraestructura será automatizada utilizando Terraform. A continuación, se mencionan las principales ventajas y características de cada una de estas tecnologías.

AWS es una plataforma cloud que ofrece servicios de infraestructura como: poder de computo, almacenamiento, bases de datos, etc., lo cual es ideal para desplegar el sistema de información de una manera ágil y a un costo menor.

---

[10] Carlos Azaustre, “Desarrollo Full Stack JavaScript”, 2014 [En línea]. Disponible: https://carlosazaustre.es/desarrollo-full-stack-javascript-tambien-conocido-como-mean/

[11] Blog.faztweb.com, “Curso de Express.js, el Framework de Nodejs | ¿Qué es Express?”, 2017 [En línea]. Disponible: http://blog.faztweb.com/2017/09/expressjs-el-framework-de-nodejs-que-es.html

Las principales ventajas y beneficios de la informática en la nube de AWS son:

- **Cambiar los gastos de inversiones en capital por gastos variables.** En lugar de tener que realizar una cuantiosa inversión en centro de datos y servidores antes de saber qué uso se le va a dar, se puede utilizar informática en la nube y pagar únicamente cuando se consuman recursos informáticos, y solamente en función del consumo realizado[4].

- **Beneficio de la economía de escala masiva.** Mediante el uso de la informática en la nube, se podrá reducir los costos variables que tendría los recursos. En la nube se suman los consumos realizados por cientos de miles de usuarios. De esta forma, los proveedores, como Amazon Web Services, pueden aplicar mayores economías de escala que se traducen en precios más bajos por el consumo realizado[4].

- **Dejar de adivinar capacidades.** Olvidar de tener que adivinar las necesidades de capacidad de la infraestructura. Al tomar una decisión respecto a la capacidad antes de implementar una aplicación, a menudo se acaba por acumular recursos caros y ociosos o se descubre que se dispone de una capacidad limitada. Con la informática en la nube, estos problemas desaparecen. Se podrá obtener acceso a los recursos que se necesite y aumentar o reducir la capacidad con unos pocos minutos de aviso[4].

- **Aumentar la velocidad y la agilidad.** En un entorno de informática en la nube, la disponibilidad de nuevos recursos está en todo momento a un simple clic del mouse. Esto significa que se puede reducir el tiempo que dichos recursos tardan en estar disponibles para los desarrolladores de semanas a cuestión de minutos. El resultado es un aumento espectacular de la agilidad de la organización, ya que se reduce notablemente el tiempo y los costos necesarios para hacer experimentos y desarrollar aplicaciones[4].

- **Dejar de gastar dinero en operar y mantener centros de datos.** Centrarse en proyectos que hagan destacar el negocio, en lugar de hacerlo en la infraestructura. La informática en la nube permite centrarse en los propios clientes, en lugar de la formidable tarea de instalar servidores en bastidores, apilarlos y proporcionarles electricidad[4].

- **Ser mundial en minutos.** Implementar su aplicación de forma sencilla en múltiples regiones alrededor del mundo con solo unos clics. De esta forma, puede ofrecer una menor latencia y una mejor experiencia a los clientes de forma sencilla y con un costo mínimo[4].

Por otro lado, **Terraform** es una herramienta de orquestación de código abierto desarrollado por Hashicorp que nos permite definir la infraestructura como código, esto quiere decir que es posible escribir en un fichero de texto la definición de la infraestructura usando un lenguaje de programación declarativo y simple[12].

Algunos de los beneficios que ofrece Terraform son:

- Administra infraestructuras considerablemente grandes o puede administrar una sola aplicación.

- No se limita a un proveedor en específico, tiene soporte para AWS, Digital Ocean, Microsoft Azure, VMware vSphere, Google Cloud Platform, etc. Para más detalle visitar el enlace: https://www.terraform.io/docs/providers/.

- Proporciona una sintaxis simple y unificada que permite administrar casi cualquier recurso en lugar de requerir que se utilicen herramientas independientes para cada plataforma y servicio.

- Las configuraciones pueden ser compartidas y reutilizables.

- El modelo de su centro de datos puede ser versionado, de esta forma es más sencillo observar el progreso de nuestro servicio y controlar los cambios[12].

6.1.2. MARCO CONCEPTUAL

Antes de poder abordar el desarrollo de un sistema como el que se ha propuesto, es importante dominar varios conceptos como son:

Cloud Computing: La informática en la nube es la entrega bajo demanda de potencia informática, almacenamiento en base de datos, aplicaciones y otros recursos de TI a través de internet como un sistema de precios basado en el consumo realizado. Una plataforma de servicios en la nube, como Amazon Web Services, es propietaria y responsable del mantenimiento del hardware conectado en red necesario para dichos servicios de aplicaciones, mientras que el cliente se dedica a provisionar lo que necesite por medio de una aplicación web[4].

[12] OpenWebinars.net, “Por qué usar Terraform?”, 2017. [En línea]. Disponible en: https://openwebinars.net/blog/por-que-usar-terraform/

Infraestructura tecnológica: Es el conjunto de hardware y software sobre el que se asientan los diferentes servicios que las empresas necesitan tener en funcionamiento para poder llevar a cabo toda su actividad[13].

Arquitectura tecnológica: Es el diseño conceptual y la estructura operacional de un sistema o conjunto de sistemas[14].

Infraestructura como servicio (IaaS): La Infraestructura como servicio, que a veces se abrevia a IaaS, contiene los bloques de creación fundamentales para la TI en la nube. Por lo general, proporciona acceso a las características de redes, a los equipos (virtuales o en software dedicado) y al espacio de almacenamiento de datos. La Infraestructura como servicio le proporciona el mayor nivel de flexibilidad y control de la administración en torno a sus recursos de TI y guarda el mayor parecido con los recursos de TI existentes con los que muchos departamentos de TI y desarrolladores están familiarizados[15].

Plataforma como servicio (PaaS): Las Plataformas como servicio eliminan la necesidad de las compañías de administrar la infraestructura subyacente (normalmente hardware y sistemas operativos) y le permiten centrarse en la implementación y la administración de sus aplicaciones. Esto contribuye a mejorar su eficacia, pues no tiene que preocuparse del aprovisionamiento de recursos, la planificación de la capacidad, el mantenimiento de software, los parches ni ninguna de las demás arduas tareas que conlleva la ejecución de su aplicación[15].

Software como servicio (SaaS): El Software como servicio le proporciona un producto completo que el proveedor del servicio ejecuta y administra. En la mayoría de los casos, quienes hablan de Software como servicio en realidad se refieren a aplicaciones de usuario final. Con una oferta de SaaS, no tiene que pensar en cómo se mantiene el servicio ni en cómo se administra la infraestructura subyacente. Solo tiene que preocuparse de cómo utilizar el software concreto. Un ejemplo común de una aplicación de SaaS es un programa de email en la Web que le permite enviar y recibir mensajes sin tener que administrar la agregación de características ni mantener los servidores y los sistemas operativos en los que se ejecuta el programa de email[15].

Amazon Web Services (AWS): La nube de AWS proporciona un amplio conjunto de servicios de infraestructura, como potencia de cómputo, opciones de almacenamiento, redes y bases de datos, ofertados como una utilidad: bajo demanda, disponibles en cuestión de segundos y pagando solo por lo que utiliza[4]. Es una de las ofertas internacionales más importantes de la computación en la

---

[13] Grupoeducare.com, "Infraestructura Tecnológica", 2019. [Online]. Available: https://www.grupoeducare.com/dcempresarial/index.php/2-principal/38-infraestructura-tecnologica.
[14] Ref
[15] Amazon Web Services, Inc. “Tipos de Cloud computing”, 2019. [En línea]. Available: https://aws.amazon.com/es/types-of-cloud-computing/.

nube y compite directamente contra servicios como Microsoft Azure y Google Cloud Platform. Es considerado como un pionero en este campo.

Amazon Elastic Compute Cloud (EC2): Amazon EC2 es un servicio web que proporciona capacidad informática en la nube segura y de tamaño modificable. Está diseñado para facilitar a los desarrolladores el uso de la informática en la nube a escala de la Web[16].

La sencilla interfaz de servicios web de Amazon EC2 permite obtener y configurar la capacidad con una fricción mínima. Proporciona un control completo sobre los recursos informáticos y puede ejecutarse en el entorno informático acreditado de Amazon. Amazon EC2 reduce el tiempo necesario para obtener y arrancar nuevas instancias de servidor en cuestión de minutos, lo que permite escalar rápidamente la capacidad, ya sea aumentándola o reduciéndola, según cambien sus necesidades. Amazon EC2 cambia el modelo económico de la informática, ya que solo tendrá que pagar por la capacidad que realmente utilice. Amazon EC2 les brinda a los desarrolladores las herramientas necesarias para crear aplicaciones resistentes a errores y para aislarlas de los casos de error comunes[16].

Amazon Simple Storage Services (S3): Amazon S3 es un servicio de almacenamiento de objetos creado para almacenar y recuperar cualquier volumen de datos desde cualquier ubicación: sitios web y aplicaciones móviles, aplicaciones corporativas y datos de sensores o dispositivos IoT. Está diseñado para ofrecer una durabilidad del 99,999999999% y almacena datos para millones de aplicaciones utilizadas por líderes de mercados de todas las industrias. S3 ofrece capacidades de conformidad y seguridad integrales que cumplen con los requisitos normativos más estrictos. Ofrece a los clientes mayor flexibilidad para administrar datos en relación con las actividades de optimización de costos, control de acceso y conformidad. S3 ofrece la funcionalidad de consulta en el lugar, lo que le permite ejecutar análisis eficientes directamente en los datos en reposo en S3. Además, Amazon S3 es la plataforma de almacenamiento de mayor compatibilidad disponible, ya que se integra con la mayoría de las soluciones de terceros, socios integradores de sistemas y otros servicios de AWS[17].

Amazon Elastic Block Store (EBS): Amazon EBS proporciona volúmenes de almacenamiento de bloques persistentes para utilizar con las instancias de Amazon EC2 en la nube de AWS. Cada volumen de Amazon EBS se replica automáticamente dentro de una zona de disponibilidad para protegerle frente a los errores de componentes, ofreciéndole una alta disponibilidad y durabilidad. Los

---

[16] Amazon Web Services, Inc. “Amazon EC2” 2018. [En línea]. Disponible en: https://aws.amazon.com/es/ec2/

[17] Amazon Web Services, Inc. “Amazon S3” 2018. [En línea]. Disponible en: https://aws.amazon.com/es/s3/

volúmenes de Amazon EBS ofrecen el desempeño constante y de baja latencia necesario para ejecutar sus cargas de trabajo. Con Amazon EBS, puede escalar el uso hacia arriba o hacia abajo en cuestión de minutos (todo esto pagando un precio reducido solo por lo que aprovisiona)[18].

Amazon Relational Database Service (RDS): Con Amazon RDS es sencillo configurar, utilizar y escalar una base de datos relacional en la nube. Proporciona capacidad rentable y de tamaño modificable y, al mismo tiempo, administra las tediosas tareas de administración de la base de datos, lo que le permite centrarse en sus aplicaciones y en su negocio. Amazon RDS le proporciona seis motores de bases de datos populares para elegir, incluido Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle y Microsoft SQL Server[19].

Amazon Elastic Load Balancing (ELB): Elastic Load Balancing distribuye automáticamente el tráfico de aplicaciones entrantes a través de varios destinos, tales como instancias de Amazon EC2, contenedores y direcciones IP. Puede manejar la carga variable del tráfico de su aplicación en una única zona o en varias zonas de disponibilidad. Elastic Load Balancing ofrece tres tipos de equilibradores de carga que cuentan con la alta disponibilidad, la escalabilidad automática y la seguridad necesaria para que las aplicaciones sean tolerantes a fallos[20].

AWS Auto Scaling: AWS Auto Scaling monitoriza las aplicaciones y ajusta automáticamente la capacidad para mantener un desempeño predecible y estable al menor costo posible. Con AWS Auto Scaling, resulta sencillo configurar el escalado de aplicaciones para distintos recursos en varios servicios en cuestión de minutos. Auto Scaling permite crear planes de escalado que automatizan la manera en la que diferentes recursos responden ante los cambios que se producen en la demanda. Puede optimizar la disponibilidad, los costos, o lograr un equilibrio entre ambos. AWS Auto Scaling crea todas las políticas de escalado y define los valores meta automáticamente en función de sus preferencias[21].

AWS Lambda: AWS Lambda permite ejecutar código sin aprovisionar ni administrar servidores. Con Lambda, puede ejecutar código para casi cualquier tipo de aplicación o servicio back-end – y todo sin administrar nada. Solo se tiene

---

[18] Amazon Web Services, Inc. “Amazon Elastic Block Store,” 2018. [En línea]. Disponible en: https://aws.amazon.com/es/ebs/

[19] Amazon Web Services, Inc. “Amazon Relational Database Services (RDS),” 2018. [En línea]. Disponible en: https://aws.amazon.com/es/rds/

[20] Amazon Web Services, Inc., “AWS Elastic Load Balancing”, 2018 [En línea]. Disponible en: https://aws.amazon.com/elasticloadbalancing/

[21] Amazon Web Services, Inc., “AWS Auto Scaling”, 2018 [En línea]. Disponible en: https://aws.amazon.com/es/autoscaling/.

que cargar el código. Lambda se encargará de todo lo necesario para ejecutar y escalar el código con alta disponibilidad[22].

Amazon DynamoDB: Es una base de datos no relacional que ofrece rendimiento fiable a cualquier escala. Es una base de datos completamente administrada, con varios nodos maestros y distribuida en varias regiones que ofrece una latencia estable con milisegundos de un dígito y almacenamiento caché en memoria, backup y restauración, y seguridad incorporada. DynamoDB ofrece una capacidad de respuesta estable de milisegundos de un dígito a cualquier escala[23].

Amazon CloudWatch: Es un servicio de monitoreo y administración creado para desarrolladores, operadores de sistemas, ingenieros de fiabilidad de sitio (SRE) y gerentes de TI. CloudWatch ofrece datos e información procesable para monitorear las aplicaciones, comprender cambios de rendimiento que afectan a todo el sistema y tomar acciones, optimizar el uso de recursos y lograr una vista unificada del estado de las operaciones. CloudWatch recopila datos de monitoreo y operaciones en formato de registros, métricas y eventos, lo que le ofrece una vista unificada de los recursos, las aplicaciones y los servicios de AWS que se ejecutan en servidores locales y de AWS. Se puede usar CloudWatch para definir alarmas de alta resolución, ver registros y métricas lado a lado, tomar acciones automatizadas, resolver errores y descubrir información para optimizar las aplicaciones y asegurar de que se estén ejecutando sin problemas[24].

AWS CloudTrail: Es un servicio que permite realizar regulaciones y auditorías operativas, de riesgo y conformidad en la cuenta de AWS. Con CloudTrail, se puede registrar, monitorear de manera continua y retener la actividad de la cuenta relacionada con llamadas a la API en toda su infraestructura de AWS. CloudTrail proporciona el historial de los eventos de la actividad de su cuenta de AWS, incluidas las acciones efectuadas a través de la consola de administración de AWS, los SDK de AWS, las herramientas de la línea de comandos y otros servicios de AWS. El historial de eventos simplifica el análisis de seguridad, el seguimiento de cambios de recursos y la resolución de problemas[25].

AWS Price List API: Proporciona acceso a los precios de los servicios de AWS en forma JSON y CSV en tiempo real y actualizada[26].

---

[22] Amazon Web Services, Inc., “AWS | Lambda”, 2018 [En linea]. Disponible en: https://aws.amazon.com/es/lambda/
[23] Amazon Web Services, Inc., “Amazon DynamoDB – NoSQL Cloud Database Service”, 2018 [En linea]. Disponible en: https://aws.amazon.com/dynamodb/
[24] Amazon Web Services, Inc., “Amazon CloudWatch – Cloud & Network Monitoring Services”, 2018 [En línea]. Disponible en: https://aws.amazon.com/cloudwatch/
[25] Amazon Web Services, Inc., “AWS CloudTrail”, 2018 [En línea]. Disponible en: https://aws.amazon.com/es/cloudtrail/
[26] AWS News Blog, “New – AWS Price List API”, 2018 [En línea]. Disponible en: https://aws.amazon.com/es/blogs/aws/new-aws-price-list-api/

Cotización: La cotización es el hecho de cotizar y significa establecer un precio, estimarlo o pagar una cuota. Se refiere a un documento informativo que el departamento de compras de una empresa o institución utiliza para entablar una negociación. Este documento no genera ninguna forma de registro contable. Sino que sólo pretenden colocar su precio justo a un bien o a un servicio[27].

Partner: Es un socio de una empresa, que se dedica a distribuir un Producto o Servicio. Partner se refiere a la relación que se establece entre dos empresas para obtener un mutuo beneficio. Habitualmente hay una empresa que es el proveedor y otra que actúa como comercializadora de los servicios/productos que ofrece el proveedor. Ambos son partner de una misma relación aunque con diferente estatus[28].

Infraestructura como código: Hace referencia a la incipiente práctica de utilizar scripts para configurar la infraestructura de computación en lugar de configurar las máquinas de forma manual. A veces denominada "infraestructura programable", la infraestructura como código (IaC) trata la configuración de la infraestructura exactamente como el software de programación. De hecho, ello comienza a difuminar los límites entre la escritura de aplicaciones y la creación de entornos donde se ejecutan. Las aplicaciones pueden contener scripts que crean y organizan sus propias máquinas virtuales. Esta es una parte fundamental de la informática en la nube y resulta esencial para DevOps[29].

Terraform: Terraform le permite crear de forma segura y predecible, cambiar y mejorar la infraestructura. Es una herramienta de código abierto que codifica API en los archivos de configuración declarativas que pueden ser compartidos entre los miembros del equipo, tratadas como código, modificar, revisar, y versionados[30].

Product Backlog: Es un listado de todas las tareas que se pretenden hacer durante el desarrollo de un proyecto. Todas las tareas deben listarse en el producto backlog, para que estén visibles ante todo el equipo y se pueda tener una visión panorámica de todo lo que se espera realizar[31].

---

[27] Economipedia, “Cotización,” 2018. [En línea]. Disponible en: http://economipedia.com/definiciones/cotizacion.html.
[28] Si-emprendemos (Blog), “Qué son los partners y como nos ayudan a los emprendedores,” 2014. [En línea]. Disponible en: https://si-emprendemos.blogspot.com.co/2014/01/que-son-los-partners-y-como-nos-ayudan.html.
[29] Hewlett packard enterprise, “¿Qué es la infraestructura como código?,” 2018. [En línea]. Disponible en: https://www.hpe.com/lamerica/es/what-is/infrastructure-as-code.html
[30] HASHICORP, “Cloud Infrastructure Automation,” 2018. [En línea]. Disponible en: https://www.terraform.io/.
[31] Ramos,J. Programacionymas.com, “¿Qué es el Product Backlog?”, 2019. [En línea]. Disponible en: https://programacionymas.com/blog/scrum-product-backlog

Sprint: Es una reunión corta que sirve para analizar como se ha trabajado durante la semana, revisar el por qué se han logrado o no los objetivos comprometidos en su inicio y si se han cumplido las expectativas del cliente[32].

7. DISEÑO METODOLÓGICO:

El desarrollo del proyecto consta de las siguientes fases que se ejecutarán de forma iterativa teniendo en cuenta la metodología Scrum:

Fase 1 – Planeación: Aquí se identifican los roles que se asumirán dentro del proyecto, la descripción de las épicas e historias de usuario, requerimientos no funcionales y la lista de tareas que se van a ejecutar (backlog del producto).

Fase 2 – Diseño: Incluye los diagramas de casos de uso, diagramas de actividades, diagrama de Base de datos, diseño de la arquitectura nube que se va a implementar y los mockups o bosquejos del prototipo web.

Fase 3 – Desarrollo: Exposición del backlog priorizado, descripción de las actividades de cada sprint con sus respectivos resultados del desarrollo

Fase 4 – Implementación y pruebas: Descripción del proceso de implementación en un ambiente productivo sobre la plataforma cloud de Amazon Web Services. Se realizarán pruebas unitarias a cada funcionalidad, pruebas de desempeño, pruebas de carga y pruebas de recuperación y tolerancia a fallos.

[32] Que es un sprint

8. FASE 1: PLANEACIÓN

En este capítulo se describe la fase de planeación realizada del proyecto basada en la metodología scrum donde incluye: la identificación del equipo Scrum, épicas, historias de usuario, requerimientos no funcionales y backlog del producto.

8.1. IDENTIFICACIÓN DE ROLES SCRUM

Durante el transcurso de este proyecto, se identifican los siguientes roles:

- **Product Owner:** El autor de este proyecto.
- **Scrum Master:** Tutor de este proyecto.
- **Equipo Scrum:** El autor y responsable de este proyecto encargado en su desarrollo, implementación y conclusión.

8.2. ÉPICAS

Una épica es vista como una historia de usuario y puede ser de alta complejidad, para realizarla es posible que se requiera un esfuerzo grande y no puede ser cubierta en un solo sprint. Una épica por su gran tamaño puede ser difícil de estimar y de abordar, así que podemos descomponerla en varias historias. La implementación de una épica por lo general suele llevar dos o más sprints.[33]

A continuación, se presentan las épicas contempladas para este proyecto:

[33] Diegoacosta.net, “Scrum con Jira – Relación entre Épicas, Historias y Tareas Técnicas. | | Blog Personal Diego Acosta", 2019. [en línea]. Disponible en: http://diegoacosta.net/blog/2017/12/10/scrum-con-jira-relacion-entre-epicas-historias-y-tareas-tecnicas/

8.2.1. ÉPICA E1

ÉPICA			
Numero:	E1	**Prioridad:**	Alta (1)
Nombre de la épica:		Información página web y gestión de cuenta	
Responsable:		Josdin Daniel Pinzón Q.	
Descripción: Como usuario quiero poder consultar públicamente la información del sistema para conocer los beneficios, un video de demostración de su uso y un formulario de contacto. A su vez, me gustaría poder crear una cuenta de forma gratuita para hacer uso del sistema.			

Tabla 2. Épica E1: Información página web y gestión de cuenta.

8.2.2. ÉPICA E2

ÉPICA			
Numero:	E2	**Prioridad:**	Media (2)
Nombre de la épica:		Consulta de plantillas predeterminadas	
Responsable:		Josdin Daniel Pinzón Q.	
Descripción: Como usuario me gustaría consultar las plantillas disponibles para tener diferentes opciones de diseño de la tabla de cotización que va a generar el sistema. Estas opciones de diseño implican diferente tamaño, estilo y color.			

Tabla 3. Épica E2: Consulta de plantillas predeterminadas.

8.2.3. ÉPICA E3

ÉPICA			
Numero:	E3	**Prioridad:**	Alta (1)
Nombre de la épica:		Formulario de cotización	
Responsable:		Josdin Daniel Pinzón Q.	
Descripción: Como usuario quiero tener un formulario dinámico donde pueda elegir qué servicios deseo cotizar en cuanto a servidores virtuales y almacenamiento, a su vez deseo que este formulario se guarde en la base de datos en caso de que se requiera una modificación de la cotización.			

Tabla 4. Épica E3: Formulario de cotización.

8.2.4. ÉPICA E4

ÉPICA			
Numero:	E4	**Prioridad:**	Media (2)
Nombre de la épica:		Gestión y configuración de impuestos	
Responsable:		Josdin Daniel Pinzón Q.	
Descripción: Como usuario administrador deseo poder crear y gestionar los diferentes porcentajes de los impuestos que puedan aplicar en las cotizaciones en diferentes países, de esta forma puedo tener control de aplicar o no aplicar impuestos sobre las cotizaciones que se vayan a generar.			

Tabla 5. Épica E4: Gestión y configuración de impuestos.

8.2.5. ÉPICA E5

ÉPICA			
Numero:	E5	**Prioridad:**	Alta (1)
Nombre de la épica:		Gestión de usuarios y permisos	
Responsable:		Josdin Daniel Pinzón Q.	
Descripción: Como usuario administrador deseo poder crear usuarios para poder brindar acceso a mi cuenta y asignar los permisos correspondientes.			

Tabla 6. Épica E5: Gestión de usuarios y permisos.

8.2.6. ÉPICA E6

ÉPICA			
Numero:	E6	**Prioridad:**	Baja (3)
Nombre de la épica:		Validación y ventanas de información	
Responsable:		Josdin Daniel Pinzón Q.	
Descripción: Como usuario necesito un procedimiento de validación de datos ingresados y ventanas de información al ejecutar acciones dentro del sistema para visualizar los resultados de las acciones ejecutadas.			

Tabla 7. Épica E6: Validación y ventajas de información

8.2.7. ÉPICA E7

ÉPICA			
Numero:	E7	**Prioridad:**	Baja (3)
Nombre de la épica:		Accesibilidad y Facilidad	
Responsable:		Josdin Daniel Pinzón Q.	
Descripción: Como usuario deseo un sistema totalmente web muy sencillo para que su uso sea muy intuitivo y fácil de usar. No deseo módulos complicados que inicialmente, no se necesite la lectura de manuales para poder usarlo. Requiero que el sistema se encuentre en idioma Universal Ingles para que pueda ser accedido y utilizado desde cualquier lugar del mundo.			

Tabla 8. Épica E7: Accesibilidad y Facilidad.

8.2.8. ÉPICA E8

ÉPICA			
Numero:	E8	**Prioridad:**	Alta (1)
Nombre de la épica:		Infraestructura	
Responsable:		Josdin Daniel Pinzón Q.	
Descripción: Como usuario deseo que el sistema sea desplegado sobre un ambiente productivo utilizando infraestructura de Amazon Web Services. Esta infraestructura debe ser altamente disponible, confiable y escalable que soporte altas cargas de demanda y pueda recuperarse en caso de desastre.			

Tabla 9. Épica E8: Infraestructura

8.3. HISTORIAS DE USUARIO

Una historia de usuario es la representación de una necesidad de negocio que puede ser implementada en un sprint y por tanto aporta valor al producto. Al final del Sprint la historia añade una nueva funcionalidad al producto y puede ser seleccionada para pasar a producción.[33]

En el backlog pueden existir historias de usuario sueltas o historias asociadas a épicas, ambas deben poderse terminar en un sprint, de lo contrario, deben ser descompuestas en historias de usuario mucho más pequeñas.[33]

A continuación, se enumeran las historias de usuario:

8.3.1. HISTORIA DE USUARIO H1

<table>
<tr><th colspan="6">Historia de usuario</th></tr>
<tr><td>Numero:</td><td>H1</td><td>Usuario:</td><td>Público</td><td>Épica:</td><td>E1</td></tr>
<tr><td colspan="2">Nombre de la historia:</td><td colspan="4">Descripción del sistema de información</td></tr>
<tr><td>Prioridad:</td><td>Alta (1)</td><td>Riesgo:</td><td colspan="3">Bajo (3)</td></tr>
<tr><td colspan="2">Responsable:</td><td colspan="4">Josdin Daniel Pinzón Q.</td></tr>
<tr><td colspan="6">Descripción:
El sistema contará con una página web principal donde cualquier usuario podrá consultar un detalle descriptivo de las funcionalidades que ofrece el sistema.</td></tr>
<tr><td colspan="6">Lista de tareas:
● Crear la sección “Welcome” en la página web.
● Describir la funcionalidad del sistema.</td></tr>
<tr><td colspan="6">Criterios de Evaluación:
● La descripción debe ser clara y concisa.</td></tr>
</table>

Tabla 10. Historia de Usuario H1: Descripción

8.3.2. HISTORIA DE USUARIO H2

Historia de usuario					
Numero:	H2	**Usuario:**	Público	**Épica:**	E1
Nombre de la historia:		Características del sistema de información			
Prioridad:	Alta (1)	**Riesgo:**	Bajo (3)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema contará con una página web principal donde cualquier usuario podrá visualizar las características del sistema de información.					
Lista de tareas: ● Crear la sección “Features” en la página web. ● Crear mínimo 6 ventajas. ● Visualizar en imágenes cada una de las características que ofrece el sistema de información.					
Criterios de Evaluación: ● Incluir una imagen por cada característica ● Incluir una descripción corta de cada característica.					

Tabla 11. Historia de Usuario H2: Características

8.3.3. HISTORIA DE USUARIO H3

Historia de usuario					
Numero:	H3	**Usuario:**	Público	**Épica:**	E1
Nombre de la historia:		Video demo del sistema de información			
Prioridad:	Baja (3)	**Riesgo:**	Bajo (3)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema contará con una página web principal donde cualquier usuario podrá reproducir un video de demostración de cómo usar la aplicación y visualizando sus características.					
Lista de tareas: ● Crear la sección “Demo” en la página web. ● Crear un video creativo donde se muestre un ejemplo del uso del sistema de información.					
Criterios de Evaluación: ● Video de alta definición, sonido claro y no más de 3 minutos.					

Tabla 12. Historia de Usuario H3: Video demo

8.3.4. HISTORIA DE USUARIO H4

<table>
<tr><th colspan="6">Historia de usuario</th></tr>
<tr><td>Numero:</td><td>H4</td><td>Usuario:</td><td>Público</td><td>Épica:</td><td>E1</td></tr>
<tr><td colspan="2">Nombre de la historia:</td><td colspan="4">Formulario de contacto</td></tr>
<tr><td>Prioridad:</td><td>Alta (1)</td><td>Riesgo:</td><td colspan="3">Bajo (3)</td></tr>
<tr><td colspan="2">Responsable:</td><td colspan="4">Josdin Daniel Pinzón Q.</td></tr>
<tr><td colspan="6">Descripción:
El sistema contará en la página web un formulario de contacto para atender cualquier solicitud de los usuarios públicos.</td></tr>
<tr><td colspan="6">Lista de tareas:
● Crear la sección “Contact us” en la página web.
● Crear un formulario de contacto con los siguientes campos:
− Nombre, correo electrónico, asunto, descripción.</td></tr>
<tr><td colspan="6">Criterios de Evaluación:
● Validación del formulario de contacto en sus tipos de datos:
− Nombre: String
− Correo electrónico: String email
− Asunto: String
− Descripción: String
● Una vez que el usuario diligencia el formulario, se envía una notificación al correo electrónico josdindaniel@gmail.com</td></tr>
</table>

Tabla 13. Historia de Usuario H4: Formulario de contacto.

8.3.5. HISTORIA DE USUARIO H5

<table>
<tr><th colspan="6">Historia de usuario</th></tr>
<tr><td>Numero:</td><td>H5</td><td>Usuario:</td><td>Publico</td><td>Épica:</td><td>E1</td></tr>
<tr><td colspan="2">Nombre de la historia:</td><td colspan="4">Creación de cuenta</td></tr>
<tr><td>Prioridad:</td><td>Alta (1)</td><td>Riesgo:</td><td colspan="3">Medio (2)</td></tr>
<tr><td colspan="2">Responsable:</td><td colspan="4">Josdin Daniel Pinzón Q.</td></tr>
<tr><td colspan="6">Descripción:
El sistema debe tener un formulario de creación de cuenta para nuevos usuarios, el cual debe solicitar los siguientes datos: Nombres, Apellidos, Correo electrónico, Compañía, País, Contraseña, Confirmar contraseña.</td></tr>
<tr><td colspan="6">Lista de tareas:
● Crear un formulario con los campos respectivos para la creación de cuentas dentro del sistema.</td></tr>
<tr><td colspan="6">Criterios de Evaluación:
● Se debe validar que el correo electrónico no se encuentre registrado en la base de datos, de lo contrario, se muestra un mensaje indicando que ya existe una cuenta con el email ingresado.
● La contraseña debe almacenarse encriptada en la base de datos.
● Validación del formulario de creación de cuenta en sus tipos de datos:
− Nombres: String
− Apellidos: String
− Correo electrónico: String email
− Compañía: String
− País: Lista seleccionable
− Contraseña: String oculto con asteriscos, debe contener mínimo 8 caracteres, una letra mayúscula, una letra minúscula, un número o un carácter especial.
− Confirmar contraseña: String oculto con asteriscos, debe coincidir con la contraseña inicial.
● Una vez validada la información ingresada, se debe almacenar en la base de datos.</td></tr>
</table>

Tabla 14. Historia de Usuario H5: Creación de cuenta

8.3.6. HISTORIA DE USUARIO H6

Historia de usuario					
Numero:	H6	**Usuario:**	Usuario	**Épica:**	E1
Nombre de la historia:		Login de cuenta			
Prioridad:	Alta (1)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe tener un formulario de login para la autenticación de usuarios que ya están creados y registrados en la base de datos, lo cual debe solicitar los siguientes datos: − Correo electrónico − Contraseña					
Lista de tareas: ● Crear un formulario con los campos respectivos para la autenticación de cuentas dentro del sistema. ● Crear un link y un formulario para la recuperación de la contraseña.					
Criterios de Evaluación: ● El sistema debe validar que los datos de email y contraseña son válidos. ● Se debe validar que el correo electrónico se encuentre registrado y concuerde con la contraseña. En caso de que el usuario olvide la contraseña, tendrá la opción de “Olvidé la contraseña” donde el sistema le enviará un link para modificar la contraseña. La nueva contraseña debe ser encriptada y almacenada en la base de datos					

Tabla 15. Historia de Usuario H6: Login

8.3.7. HISTORIA DE USUARIO H7

Historia de usuario					
Numero:	H7	**Usuario:**	Usuario	**Épica:**	E1
Nombre de la historia:		Configuración de cuenta			
Prioridad:	Media (2)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema proporciona visualización de los datos del usuario y preferencias de la cuenta, y debe permitir modificarlos.					
Lista de tareas: ● Crear un formulario con los siguientes datos de usuario: − Nombres (Editable) − Apellidos (Editable) − Correo electrónico (No editable) − Compañía (No editable) − País (Editable) − Contraseña (Editable) − Confirmar contraseña (Editable)					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● Los dos campos de contraseña deben coincidir en caso de que el usuario necesite cambiar su contraseña.					

Tabla 16. Historia de Usuario H7: Configuración cuenta

8.3.8. HISTORIA DE USUARIO H8

<table>
<tr><th colspan="6">Historia de usuario</th></tr>
<tr><td>Numero:</td><td>H8</td><td>Usuario:</td><td>Usuario</td><td>Épica:</td><td>E2</td></tr>
<tr><td colspan="2">Nombre de la historia:</td><td colspan="4">Listado de plantillas</td></tr>
<tr><td>Prioridad:</td><td>Media (2)</td><td>Riesgo:</td><td colspan="3">Medio (2)</td></tr>
<tr><td colspan="2">Responsable:</td><td colspan="4">Josdin Daniel Pinzón Q.</td></tr>
<tr><td colspan="6">Descripción:
El sistema muestra una lista de plantillas disponibles</td></tr>
<tr><td colspan="6">Lista de tareas:
● Crear 8 plantillas o estilos para la tabla de la cotización.
● Las plantillas estarán disponibles inicialmente con los siguientes colores:
− White
− Blue
− Green
− Red
− Black
− Orange
− Yellow
− Purple
● Esta sección solo permite ver un ejemplo de uso de cada una de las plantillas disponibles.</td></tr>
<tr><td colspan="6">Criterios de Evaluación:
● Únicamente disponible cuando el usuario se haya autenticado en el sistema.</td></tr>
</table>

Tabla 17. Historia de Usuario H8: Plantillas

8.3.9. HISTORIA DE USUARIO H9

Historia de usuario					
Numero:	H9	**Usuario:**	Usuario	**Épica:**	E3
Nombre de la historia:		Listado de cotizaciones			
Prioridad:	Alta (1)	**Riesgo:**	Alto (1)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe mostrar en una tabla el listado de las cotizaciones generadas.					
Lista de tareas: ● Crear una tabla con las siguientes columnas: − Titulo − Compañía − Fecha − Usuario − Editar − Eliminar − Ver ● La tabla mostrará los registros consultados en la base de datos de las cotizaciones generadas					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● La opción de “Ver” permite visualizar la cotización generada en PDF. ● La tabla debe permitir filtrar por título de la cotización, tener paginación y limitar el número de ítems por página.					

Tabla 18. Historia de Usuario H9: Listado de cotizaciones

8.3.10. HISTORIA DE USUARIO H10

Historia de usuario					
Numero:	H10	**Usuario:**	Usuario	**Épica:**	E3
Nombre de la historia:		Crear cotización			
Prioridad:	Alta (1)	**Riesgo:**	Alto (1)		
Responsable:		Josdin Daniel Pinzón Q.			

Descripción:
El sistema debe tener un formulario dinámico para el registro de nuevas cotizaciones

Lista de tareas:

- La cotización debe tener los siguientes campos:
  - Titulo
  - Compañía
  - Plantilla
  - Región
  - Usuario
- Crear un formulario dinámico para la elección de los servicios a cotizar:
  - Amazon EC2
  - Amazon EBS
  - Amazon S3
- Al crear una cotización el sistema debe generar un archivo pdf para visualizar el resultado.

Criterios de Evaluación:

- Únicamente disponible cuando el usuario se haya autenticado en el sistema.
- Validación del formulario dinámico en sus tipos de datos:
  <u>Amazon EC2:</u>
  - Name (String alfanumérico)
  - Quantity (Entero)
  - Usage (Entero)
  - OS (Combo)
  - Instance Type (Combo)
  - Billing Option (Combo)

  <u>Amazon EBS:</u>
  - Name (String alfanumérico)
  - Quantity (Entero)
  - Volumen Type (Combo)

- Storage (Entero)

Amazon S3:

- Name (String alfanumérico)
- Type (Combo)
- Storage (Entero)

- Se debe mostrar un mensaje avisando que se creó la cotización correctamente.
- La región por defecto donde se cotizan los servicios de AWS es US-EAST (Virginia), por ahora no se contempla otras regiones.
- Las cotizaciones deben tener un título, una fecha de creación, una plantilla asignada y el nombre de la compañía.
- El formulario debe ser dinámico, esto significa que el sistema permite agregar o eliminar campos según sea necesario.

Tabla 19. Historia de Usuario H10

8.3.11. HISTORIA DE USUARIO H11

Historia de usuario					
Numero:	H11	**Usuario:**	Usuario	**Épica:**	E3
Nombre de la historia:		Editar información cotizaciones			
Prioridad:	Alta (1)	**Riesgo:**	Alta (1)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe permitir modificar la información de las cotizaciones registradas					
Lista de tareas: ● En la misma tabla donde se listan las cotizaciones registradas, agregar una columna “Editar”, de tal forma que al hacer clic se pueda editar la información.					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● Validación del formulario dinámico en sus tipos de datos respectivos. ● Se debe mostrar un mensaje avisando que se modificó la cotización correctamente. ● En el momento de guardar los cambios, automáticamente el sistema genera una nueva cotización en PDF.					

Tabla 20. Historia de Usuario H11: Editar cotizaciones

8.3.12. HISTORIA DE USUARIO H12

Historia de usuario					
Numero:	H12	**Usuario:**	Usuario	**Épica:**	E3
Nombre de la historia:		Eliminar cotizaciones			
Prioridad:	Alta (1)	**Riesgo:**	Alta (1)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe permitir eliminar el registro de una cotización registrada					
Lista de tareas: ● En la misma tabla donde se listan las cotizaciones registradas, agregar una columna “Eliminar”, de tal forma que al hacer clic se pueda eliminar la información directamente.					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● Se debe mostrar un mensaje avisando que se eliminó la cotización correctamente.					

Tabla 21. Historia de Usuario H12: Eliminar cotizaciones

8.3.13. HISTORIA DE USUARIO H13

Historia de usuario					
Numero:	H13	**Usuario:**	Usuario	**Épica:**	E4
Nombre de la historia:		Listado de impuestos			
Prioridad:	Media (2)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe mostrar en una tabla el listado de los impuestos disponibles para ser aplicados a las cotizaciones.					
Lista de tareas: ● Crear una tabla con las siguientes columnas: − País − Impuesto − Editar − Eliminar ● La tabla mostrará los registros consultados en la base de datos de los impuestos registrados					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● La tabla debe permitir filtrar por nombre, tener paginación y limitar el número de ítems por página.					

Tabla 22. Historia de Usuario H13: Listado impuestos

8.3.14. HISTORIA DE USUARIO H14

Historia de usuario					
Numero:	H14	**Usuario:**	Usuario	**Épica:**	E4
Nombre de la historia:		Crear impuesto			
Prioridad:	Media (2)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe tener un formulario para el registro de nuevos impuestos					
Lista de tareas: ● Crear formulario para la creación de un usuario con los siguientes campos: − País − Porcentaje					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● Se debe validar que no existe un país ya asignado con un impuesto, si existe no permitirá crear un nuevo impuesto. ● Validación del formulario de contacto en sus tipos de datos: − País: Lista seleccionable − Porcentaje: Decimal ● Se debe mostrar un mensaje avisando que se creó el impuesto correctamente.					

Tabla 23. Historia de Usuario H14: Crear impuesto

8.3.15. HISTORIA DE USUARIO H15

Historia de usuario					
Numero:	H15	**Usuario:**	Usuario	**Épica:**	E4
Nombre de la historia:		Editar información impuesto			
Prioridad:	Media (2)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe permitir modificar la información de los impuestos registrados.					
Lista de tareas: ● En la misma tabla donde se listan los impuestos registrados, agregar una columna “Editar”, de tal forma que al hacer clic se pueda editar la información. ● Crear un formulario de edición de datos similar al formulario al de creación de un impuesto					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● Validación del formulario de contacto en sus tipos de datos: − País: Lista seleccionable No editable − Porcentaje: Decimal ● Se debe mostrar un mensaje avisando que se modificó el usuario correctamente.					

Tabla 24. Historia de Usuario H15: Editar usuario

8.3.16. HISTORIA DE USUARIO H16

Historia de usuario					
Numero:	H16	**Usuario:**	Usuario	**Épica:**	E4
Nombre de la historia:		Eliminar impuesto			
Prioridad:	Media (2)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe permitir eliminar el registro de un impuesto					
Lista de tareas: ● En la misma tabla donde se listan los impuestos registrados, agregar una columna “Eliminar”, de tal forma que al hacer clic se pueda eliminar la información directamente.					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● Se podrán eliminar impuestos siempre y cuando no haya cotizaciones asociadas, de lo contrario no dejará eliminar. ● Se debe mostrar un mensaje avisando que se eliminó el usuario correctamente.					

Tabla 25. Historia de Usuario H16: Eliminar impuesto

8.3.17. HISTORIA DE USUARIO H17

Historia de usuario					
Numero:	H17	**Usuario:**	Admin	**Épica:**	E5
Nombre de la historia:		Listado de usuarios			
Prioridad:	Media (2)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe mostrar en una tabla el listado de los usuarios registrados					
Lista de tareas: ● Crear una tabla con las siguientes columnas: − Nombre − Apellido − Correo electrónico − Estado − Fecha − Editar − Eliminar ● La tabla mostrará los registros consultados en la base de datos de los usuarios registrados					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● La tabla debe permitir filtrar por nombre, tener paginación y limitar el número de ítems por página.					

Tabla 26. Historia de Usuario H17: Listado usuarios

8.3.18. HISTORIA DE USUARIO H18

<table>
<tr><th colspan="6">Historia de usuario</th></tr>
<tr><td>Numero:</td><td>H18</td><td>Usuario:</td><td>Admin</td><td>Épica:</td><td>E5</td></tr>
<tr><td colspan="2">Nombre de la historia:</td><td colspan="4">Crear usuarios</td></tr>
<tr><td>Prioridad:</td><td>Media (2)</td><td>Riesgo:</td><td colspan="3">Medio (2)</td></tr>
<tr><td colspan="2">Responsable:</td><td colspan="4">Josdin Daniel Pinzón Q.</td></tr>
<tr><td colspan="6">Descripción:
El sistema debe tener un formulario para el registro de nuevos usuarios</td></tr>
<tr><td colspan="6">Lista de tareas:
● Crear formulario para la creación de un usuario con los siguientes campos:
− Nombre
− Apellido
− Correo electrónico
− Contraseña
− Confirmar contraseña
− Estado
− Rol
● Permite asignar cualquiera de los siguientes 3 roles con sus respectivos permisos:
<table>
<tr><th>Rol</th><th>Permisos</th></tr>
<tr><td>Administrador</td><td>Acceso total al sistema.</td></tr>
<tr><td>Editor</td><td>Acceso total excepto la administración de usuarios.</td></tr>
<tr><td>Visualizador</td><td>Solo consulta, no podrá editar, crear, ni eliminar.</td></tr>
</table></td></tr>
<tr><td colspan="6">Criterios de Evaluación:
● Únicamente disponible cuando el usuario se haya autenticado en el sistema.
● Se debe validar que el correo electrónico exista, si existe no permitirá crear un nuevo usuario.
● Validación del formulario de contacto en sus tipos de datos:</td></tr>
</table>

- Nombre: String
- Apellido: String
- Correo electrónico: String email
- Contraseña: Alfanumérico
- Estado: Boolean
- Permisos: Entero

- Se debe mostrar un mensaje avisando que se creó el usuario correctamente.

Tabla 27. Historia de Usuario H18: Crear usuario

8.3.19. HISTORIA DE USUARIO H19

Historia de usuario					
Numero:	H19	**Usuario:**	Admin	**Épica:**	E5
Nombre de la historia:		Editar información usuario			
Prioridad:	Media (2)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe permitir modificar la información de los usuarios registrados					
Lista de tareas: ● En la misma tabla donde se listan los usuarios registrados, agregar una columna “Editar”, de tal forma que al hacer clic se pueda editar la información. ● Crear un formulario de edición de datos similar al formulario al de creación de un usuario					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● El correo electrónico no puede modificarse a excepción de los demás campos. ● Validación del formulario de contacto en sus tipos de datos: − Nombre: String − Apellido: String − Correo electrónico: String email − Contraseña: Alfanumérico − Estado: Boolean − Permisos: Entero ● Se debe mostrar un mensaje avisando que se modificó el usuario correctamente.					

Tabla 28. Historia de Usuario H19: Editar usuario

8.3.20. HISTORIA DE USUARIO H20

Historia de usuario					
Numero:	H20	**Usuario:**	Admin	**Épica:**	E5
Nombre de la historia:		Eliminar usuarios			
Prioridad:	Media (2)	**Riesgo:**	Medio (2)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe permitir eliminar el registro de un usuario					
Lista de tareas: ● En la misma tabla donde se listan los usuarios registrados, agregar una columna “Eliminar”, de tal forma que al hacer clic se pueda eliminar la información directamente.					
Criterios de Evaluación: ● Únicamente disponible cuando el usuario se haya autenticado en el sistema. ● Se podrán eliminar usuarios siempre y cuando no haya cotizaciones asociadas, de lo contrario no dejará eliminar. ● Se debe mostrar un mensaje avisando que se eliminó el usuario correctamente.					

Tabla 29. Historia de Usuario H20: Eliminar usuarios

8.3.21. HISTORIA DE USUARIO H21

Historia de usuario					
Numero:	H21	**Usuario:**	Usuario	**Épica:**	E6
Nombre de la historia:		Avisos al usuario			
Prioridad:	Baja (3)	**Riesgo:**	Bajo (3)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe mostrar ventanas o avisos de información en cualquier acción que se ejecute al crear, modificar, editar y eliminar cualquier información					
Lista de tareas: ● Realizar pruebas e implementar ventanas de información en todas las secciones del sistema.					
Criterios de Evaluación: ● Los avisos deben contener mensajes claros y de un color que se puedan identificar fácilmente, rojo para errores, y verdes para acciones ejecutadas correctamente.					

Tabla 30. Historia de Usuario H21: Avisos de usuario

8.3.22. HISTORIA DE USUARIO H22

Historia de usuario					
Numero:	H22	**Usuario:**	Usuario	**Épica:**	E7
Nombre de la historia:		Funcionalidad fácil y sencilla			
Prioridad:	Baja (3)	**Riesgo:**	Bajo (3)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe ser muy fácil de manejar y entender					
Lista de tareas: ● Realizar pruebas en todas las secciones del sistema minimizando en lo más posible procesos complejos.					
Criterios de Evaluación: ● El sistema debe ser fácil de entender y rápido de acceder para ejecutar cualquier acción que se desee.					

Tabla 31. Historia de Usuario H22: Funcionalidad

8.3.23. HISTORIA DE USUARIO H23

Historia de usuario					
Numero:	H23	**Usuario:**	NA	**Épica:**	E8
Nombre de la historia:		Infraestructura			
Prioridad:	Alta (1)	**Riesgo:**	Alta (1)		
Responsable:		Josdin Daniel Pinzón Q.			
Descripción: El sistema debe contar con un ambiente productivo el cual consiste en aprovisionar recursos de infraestructura que garanticen altos niveles de disponibilidad, confiabilidad y escalabilidad					
Lista de tareas: ● Aprovisionar y automatizar el despliegue en un ambiente productivo con altos niveles de disponibilidad y escalabilidad sobre una infraestructura de Amazon Web Services.					
Criterios de Evaluación: ● Se debe elaborar un script de automatización para el ambiente de producción. ● La ejecución del script debe ejecutarse sin errores.					

Tabla 32. Historia de Usuario H23: Infraestructura

8.4. REQUERIMIENTOS NO FUNCIONALES

El prototipo web cuenta con las siguientes características no funcionales:

- El idioma por defecto que utilizará la aplicación es el inglés.
- El sistema deberá responder rápidamente a las peticiones generadas de los usuarios en menos de 10 segundos.
- El sistema debe ser capaz de operar adecuadamente con hasta 100 usuarios con sesiones concurrentes.
- El sistema debe ser fácil de usar, con ayudas e interfaces intuitivas.
- El ingreso al sistema estará restringido bajo contraseñas cifradas, usuarios y roles definidos. Los permisos podrán ser cambiados únicamente por el administrador de acceso a datos.
- Se podrá acceder al sistema a cualquier hora y desde cualquier lugar del mundo.
- El sistema debe respaldarse cada 24 horas. Los respaldos deben ser almacenados en una localidad segura que garantice durabilidad y disponibilidad del 99,99%.
- El sistema debe proporcionar mensajes de error que sean informativos y orientados al usuario final.

8.5. BACKLOG DEL PRODUCTO

A continuación, se lista el backlog del producto donde se visualiza las épicas con sus respectivas historias de usuario, asignado a un responsable con su respectiva prioridad y riesgo:

ÉPICA	PRIORIDAD	HISTORIA DE USUARIO	PRIORIDAD	RIESGO
E1: Información página web y gestión de cuenta	1	H1: Descripción del sistema de información	1	3
		H2: Características del sistema de información	1	3
		H3: Video demo del sistema de información	3	3
		H4: Formulario de contacto	1	3
		H5: Creación de cuenta	1	2
		H6: Login de cuenta	1	2
		H7: Configuración y preferencias de cuenta	2	2
E2: Selección de Plantillas	2	H8: Listado de plantillas	2	2

E3: Formulario de cotización	1	H9: Listado de cotizaciones	1	1
		H10: Crear cotización	1	1
		H11: Editar información cotizaciones	1	1
		H12: Eliminar cotizaciones	1	1
E4: Gestión y configuración de impuestos	2	H13: Listado de impuestos	2	2
		H14: Crear impuesto	2	2
		H15: Editar información impuesto	2	2
		H16: Eliminar impuesto	2	2
E5: Gestión de usuarios y permisos	2	H17: Listado de usuarios	2	2
		H18: Crear usuarios	2	2
		H19: Editar información usuario	2	2
		H20: Eliminar usuarios	2	2
E6: Validación y ventanas de información	3	H21: Avisos al usuario	3	3
E7: Accesibilidad y Facilidad	3	H22: Funcionalidad fácil y sencilla	3	3
E8: Ambiente de producción	1	H23: Infraestructura	1	1

Tabla 33. Backlog del producto

9. FASE 2: DISEÑO

En este capítulo se evidencia la fase de diseño del prototipo web que incluye los diagramas de casos de uso, diagramas de actividades, diagrama de Base de datos, diseño de la arquitectura nube que se va a implementar y los mockups o bosquejos del prototipo web.

9.1. DIAGRAMAS DE CASOS DE USO

Los diagramas de casos de uso representan la forma en como los actores interactúan con el sistema desde el punto de vista del usuario. En seguida se visualizan los casos de uso para el prototipo web:

9.1.1. CASO DE USO CU1

El actor "Visitante" es un usuario no autenticado que puede consultar la página web del sistema desde cualquier lugar. El actor puede consultar la descripción del sistema, consultar las ventajas que le ofrece el servicio, ver un video de demostración para que no sienta la necesidad de crear una cuenta obligatoriamente, puede diligenciar un formulario de contacto en caso de que tenga preguntas o comentarios frente al servicio y finalmente puede crear una cuenta para tener acceso al sistema.

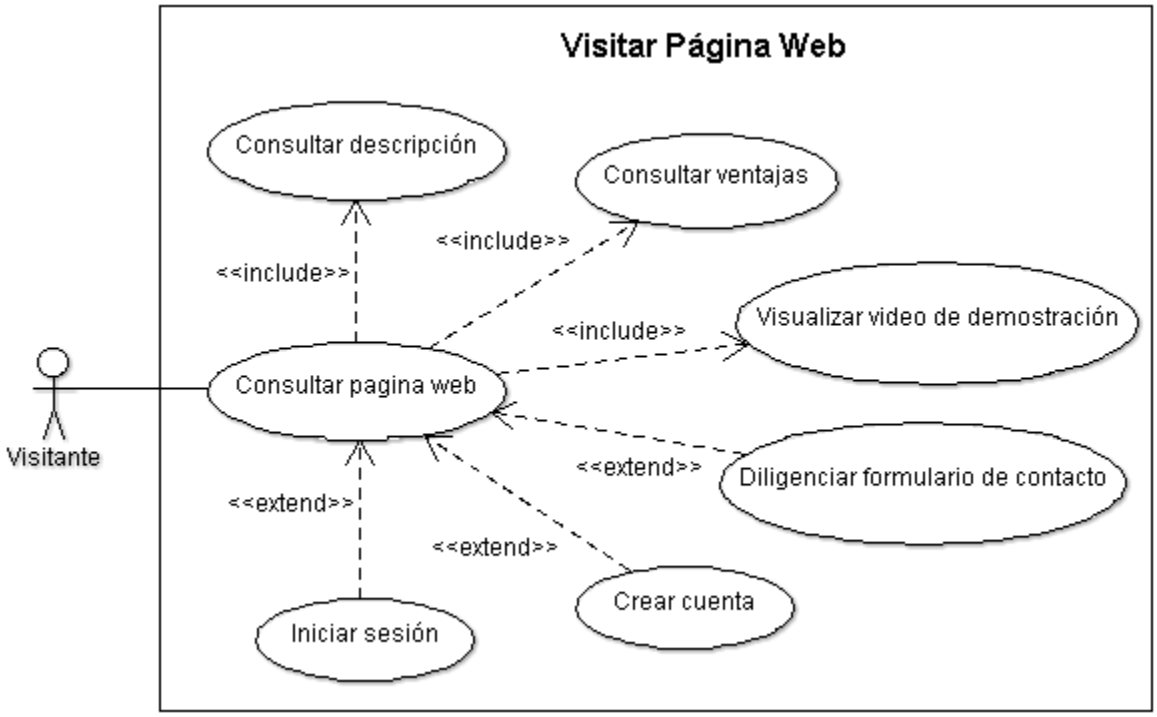


Figura 14. Caso de uso CU1: Página web

9.1.2. CASO DE USO CU2

El actor “Visitante” una vez haya creado una cuenta, se convierte en el actor “Usuario” el cual puede iniciar sesión y consultar los datos personales de la cuenta, así mismo, puede consultar los distintos módulos disponibles como el módulo de cotizaciones, el módulo de impuestos, el módulo de plantillas y el módulo de usuarios. En los siguientes casos de uso se muestra en detalle cada una de las acciones que puede realizar el usuario en cada uno de los módulos mencionados.

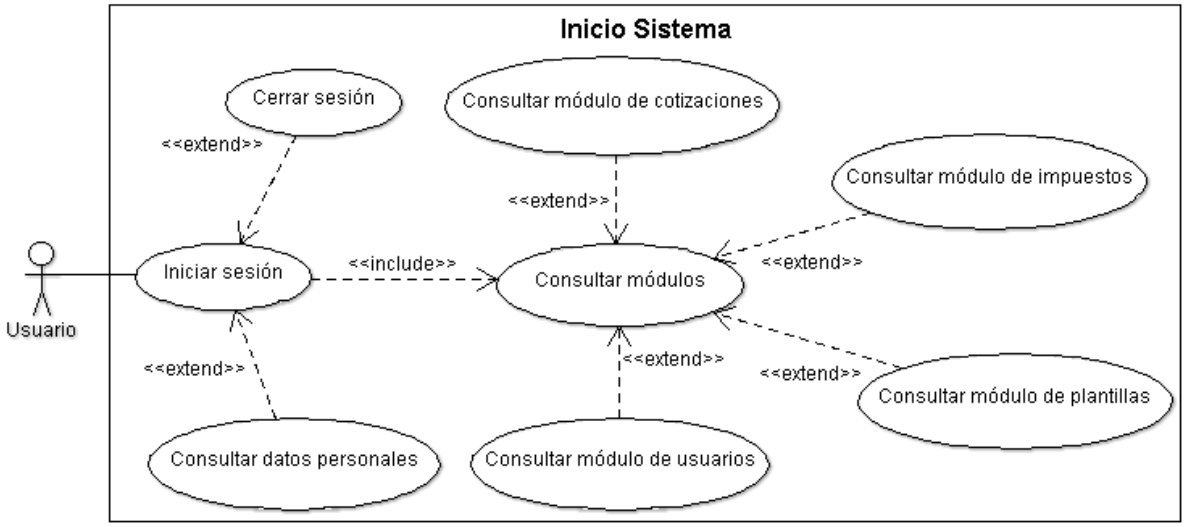


Figura 15. Caso de uso CU2: Inicio del sistema

9.1.3. CASO DE USO CU3

El usuario autenticado puede consultar y modificar sus datos personales como nombres, apellidos, correo electrónico, empresa y país, así mismo tiene la posibilidad de cambiar su contraseña.

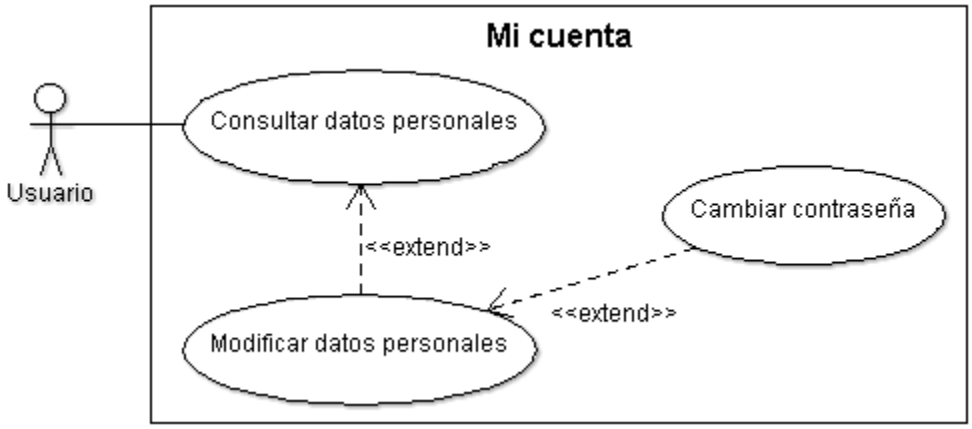


Figura 16. Caso de uso CU3: Mi cuenta

9.1.4. CASO DE USO CU4

El usuario autenticado puede acceder al módulo de plantillas en el cual puede consultar el listado de plantillas predefinidas disponibles en el sistema.

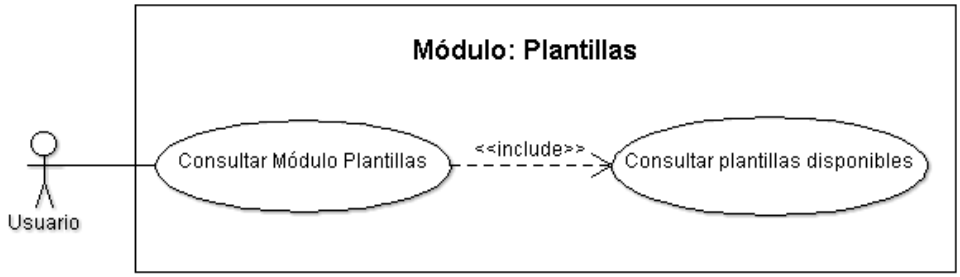


Figura 17. Caso de uso CU4: Plantilla

9.1.5. CASO DE USO CU5

El usuario autenticado puede acceder al módulo de cotización en el cual puede consultar el listado de cotizaciones generadas y guardadas en el sistema, puede realizar una búsqueda específica filtrando el nombre de la cotización, puede editarla y eliminarla, así mismo puede crear nuevas cotizaciones donde debe ingresar un título o nombre de la cotización, nombre de la empresa a quien va dirigida, asociar una plantilla de las que están predefinidas en el sistema, asociar la región de AWS, y seleccionar los servicios que se desea cotizar, posteriormente puede guardar y generar la cotización o simplemente cancelar la operación.

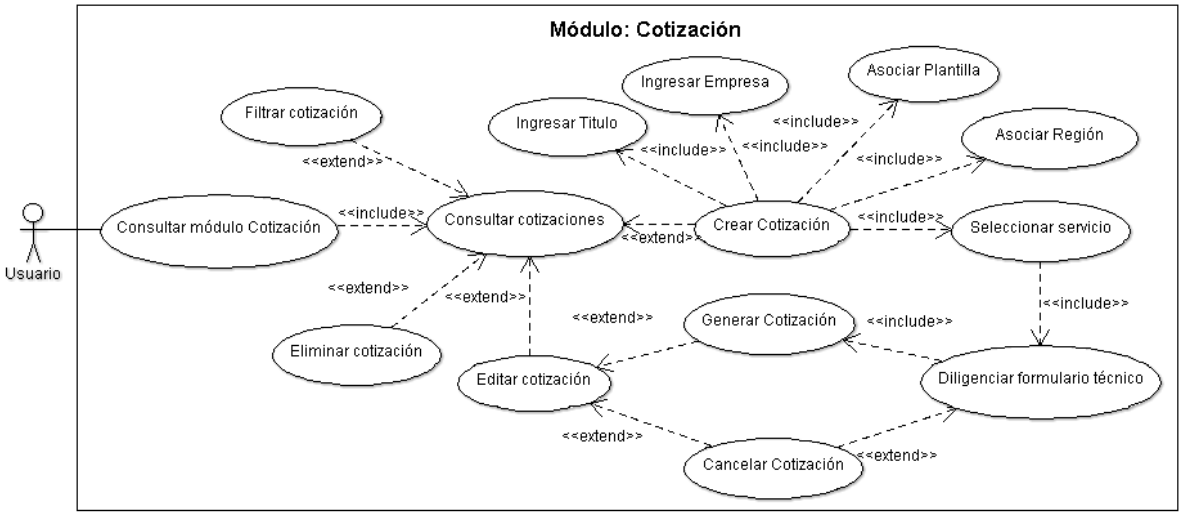


Figura 18. Caso de uso CU5: Cotización

9.1.6. CASO DE USO CU6

El usuario autenticado que ha creado la cuenta inicialmente se considera como actor “Admin”, es el único usuario que tiene acceso al Módulo usuarios, en el cual puede crear, editar y eliminar usuarios bajo la misma cuenta, puede definir un estado a los usuarios (activo o desactivado) sin la necesidad de eliminar el usuario, así mismo puede asignar permisos a usuarios como administrador, editor o visualizador.

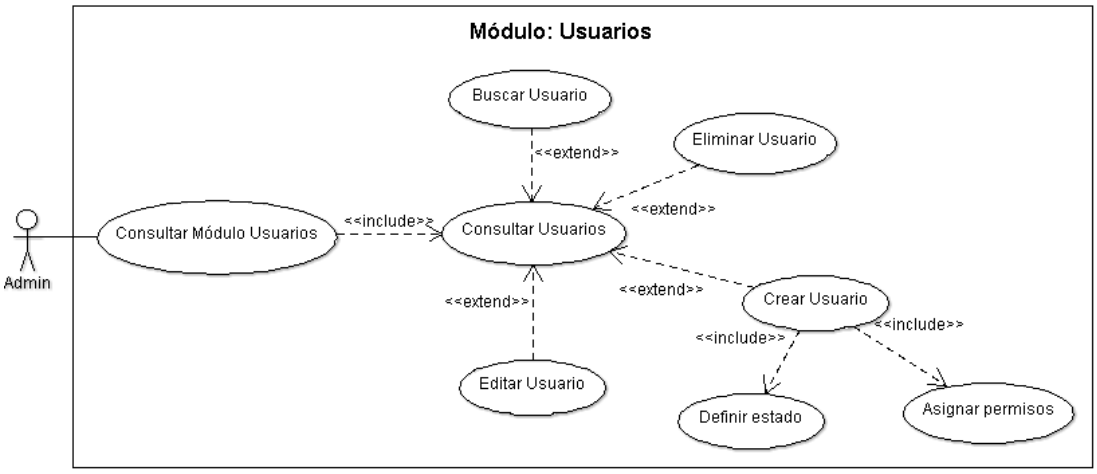


Figura 19. Caso de uso CU6: Usuarios

9.1.7. CASO DE USO CU7

El usuario autenticado puede acceder al módulo de impuestos en el cual puede crear, editar y eliminar impuestos asociados a un país.

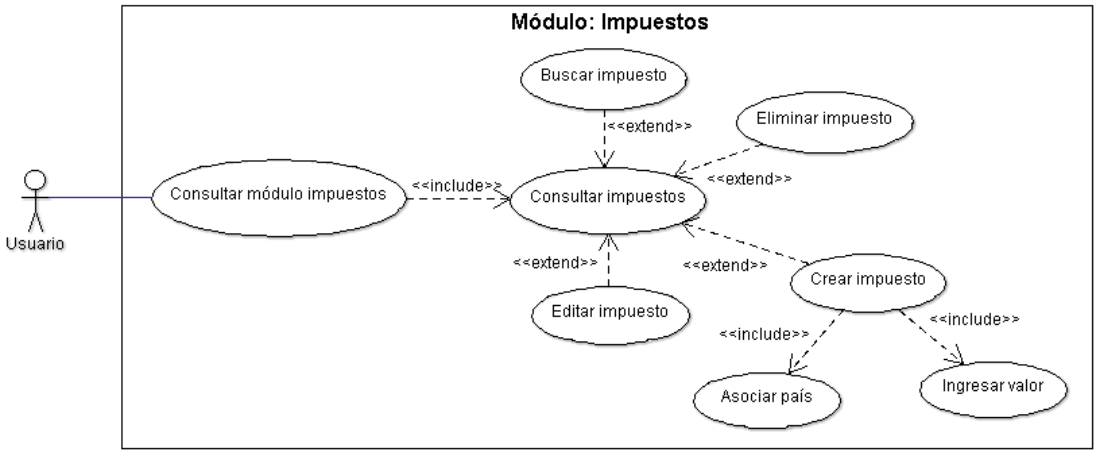


Figura 20. Caso de uso CU7: Usuarios

Eda

9.2. DIAGRAMAS DE ACTIVIDADES

Un diagrama de actividades muestra un proceso de negocio como un flujo de trabajo a través de una serie de acciones, a continuación, se visualizan los diagramas de actividades para el prototipo web:

9.2.1. ACTIVIDAD A1

La Figura 21 describe las actividades de un usuario cuando va a ingresar o iniciar sesión en el sistema.

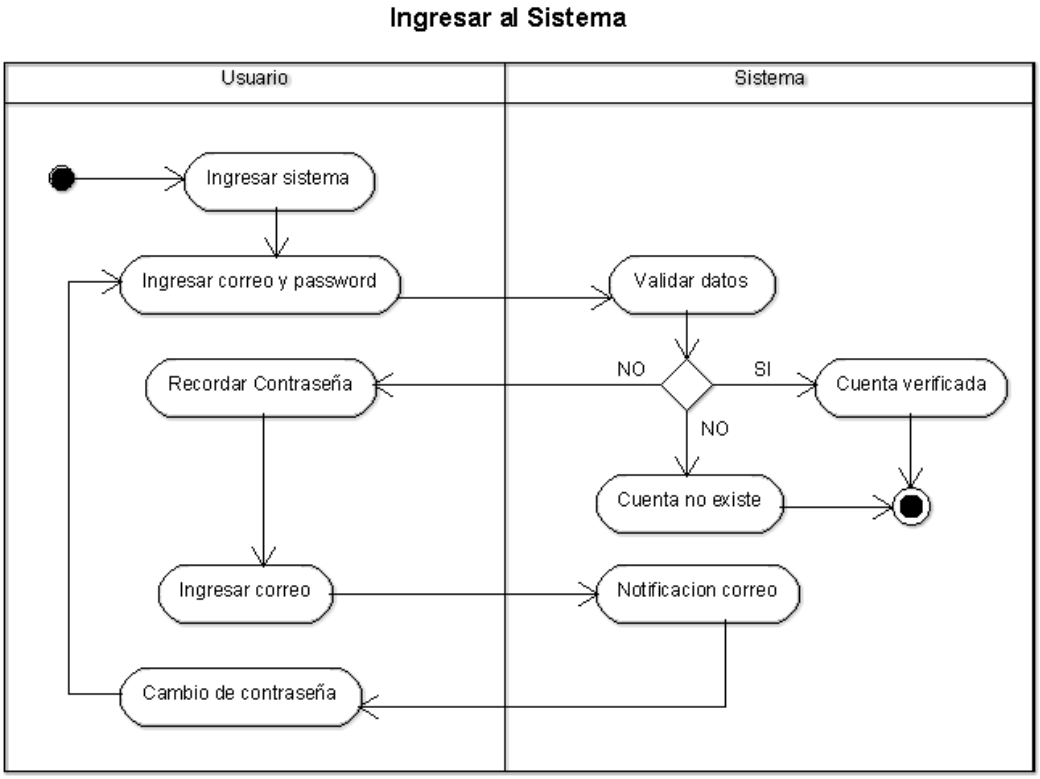


Figura 21. Actividad A1: Ingresar al sistema

9.2.2. ACTIVIDAD A2

La Figura 22 describe las actividades de un usuario cuando va a consultar y/o modificar los datos de su cuenta.

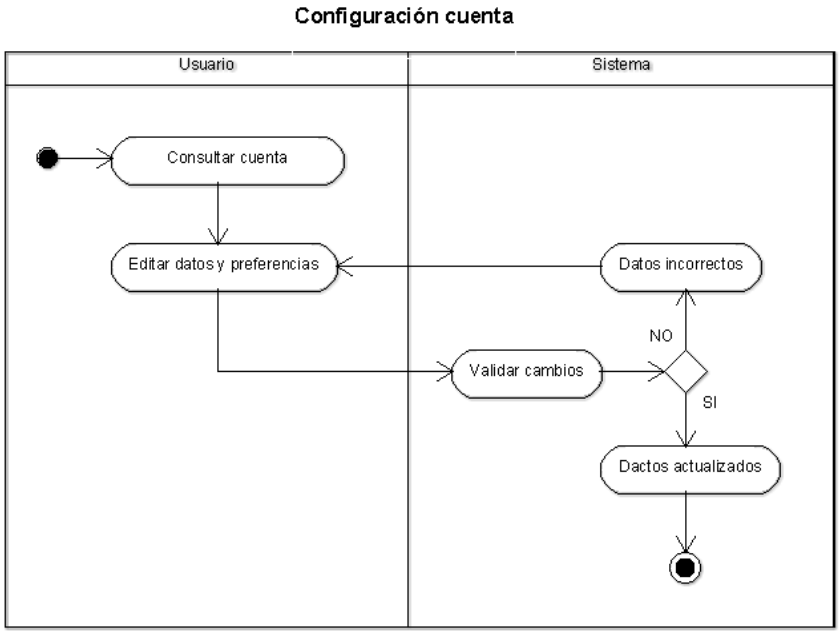


Figura 22. Actividad A2: Configuración cuenta

9.2.3. ACTIVIDAD A3

La Figura 23 describe las actividades de un usuario cuando va a crear una cotización en el sistema.

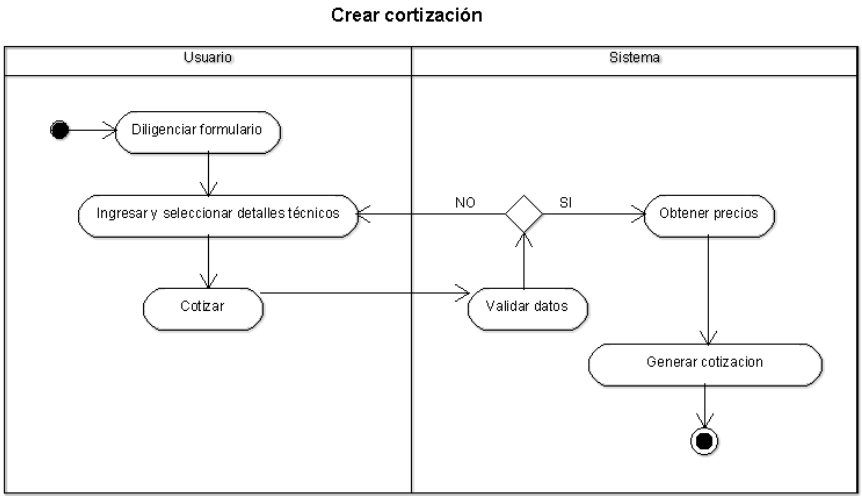


Figura 23. Actividad A3: Crear cotización

9.2.4. ACTIVIDAD A4

La Figura 24 describe las actividades de un usuario cuando va a editar una cotización en el sistema.

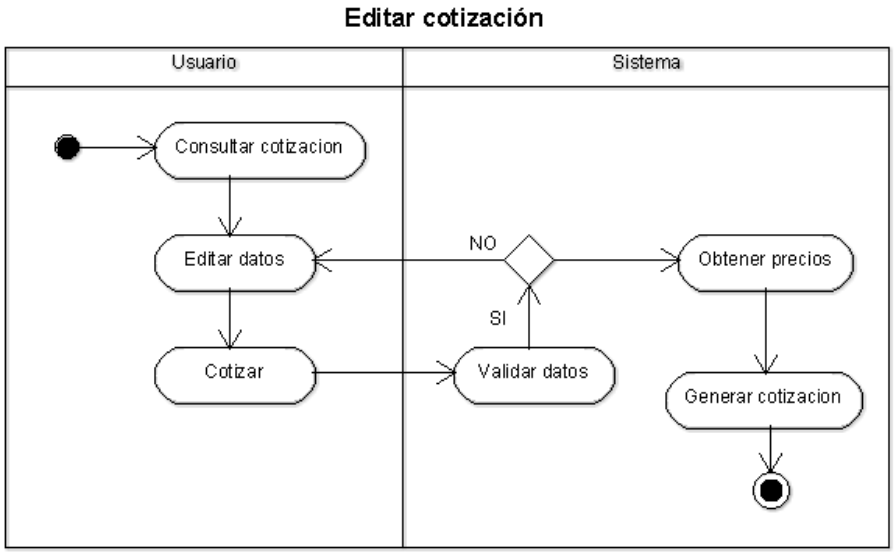


Figura 24. Actividad A4: Editar cotización

9.2.5. ACTIVIDAD A5

La Figura 25 describe las actividades de un usuario cuando va a consultar y visualizar una cotización en el sistema.

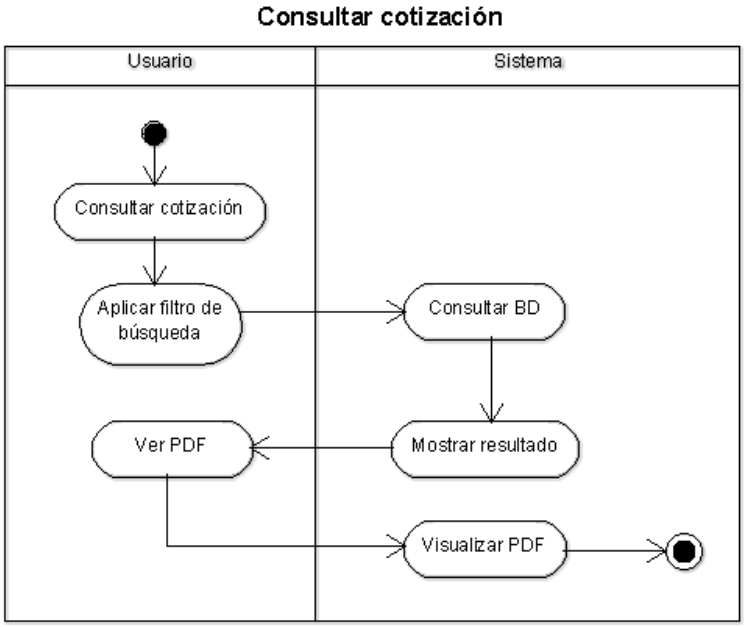


Figura 25. Actividad A5: Consultar cotización

9.2.6. ACTIVIDAD A6

La Figura 26 describe las actividades de un usuario cuando va a eliminar una cotización en el sistema.

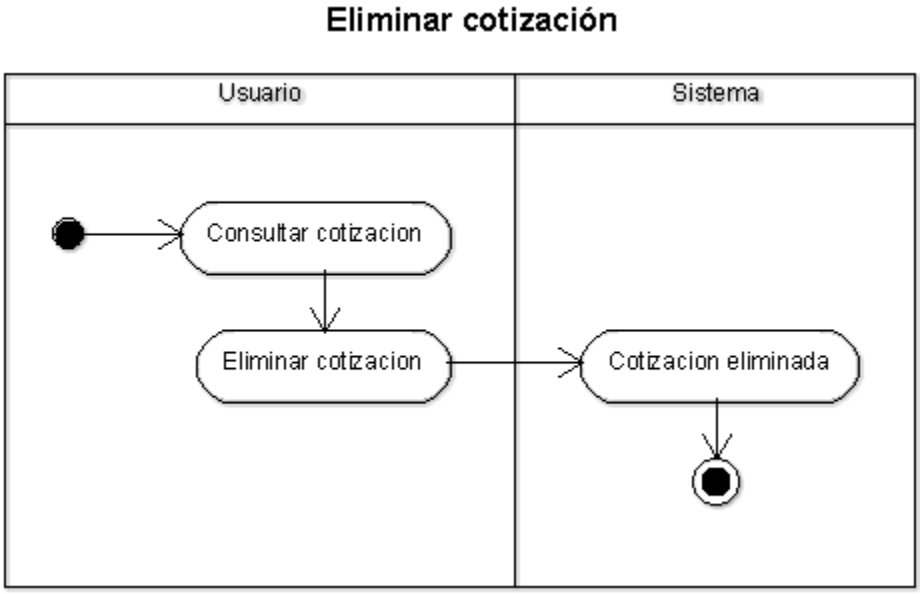


Figura 26. Actividad A6: Eliminar cotización

9.2.7. ACTIVIDAD A7

La Figura 27 describe las actividades de un administrador cuando va a crear un usuario bajo su cuenta en el sistema.

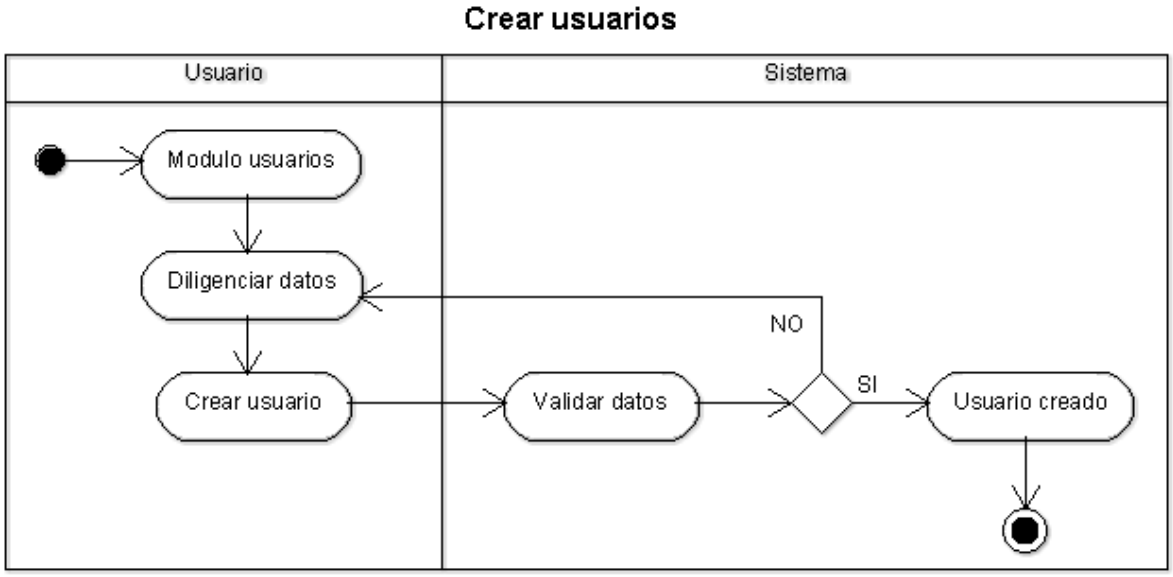


Figura 27. Actividad A7: Crear usuarios

9.2.8. ACTIVIDAD A8

La Figura 28 describe las actividades de un administrador cuando va a editar un usuario bajo su cuenta en el sistema.

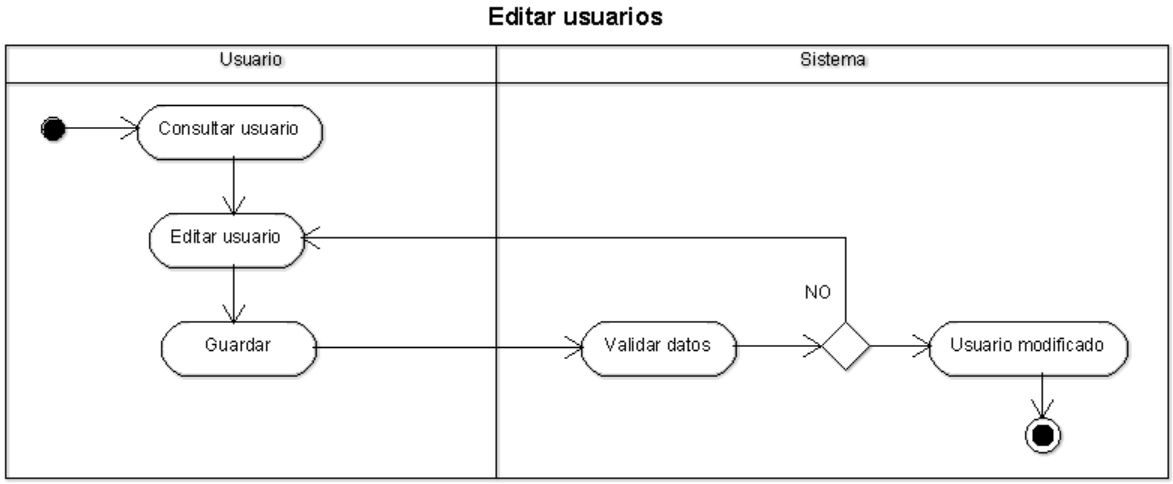


Figura 28. Actividad A8: Editar usuarios

9.2.9. ACTIVIDAD A9

La Figura 29 describe las actividades de un administrador cuando va a consultar un usuario bajo su cuenta en el sistema.

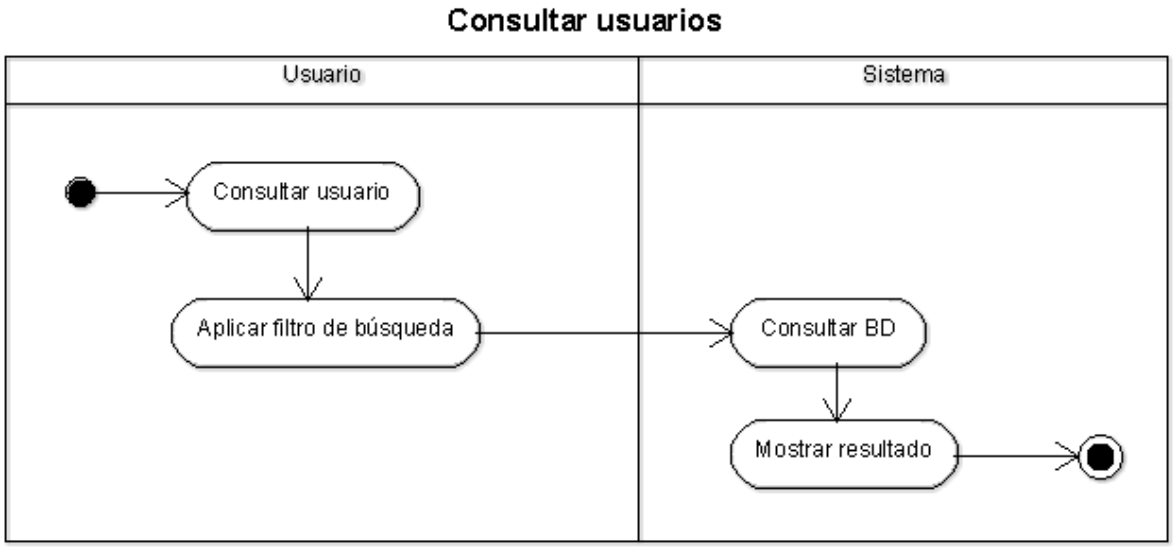


Figura 29. Actividad A9: Consultar usuarios

9.2.10. ACTIVIDAD A10

La Figura 30 describe las actividades de un administrador cuando va a eliminar un usuario bajo su cuenta en el sistema, el sistema valida que ese usuario no esté relacionado a una cotización creada, de lo contrario no permite eliminar el usuario.

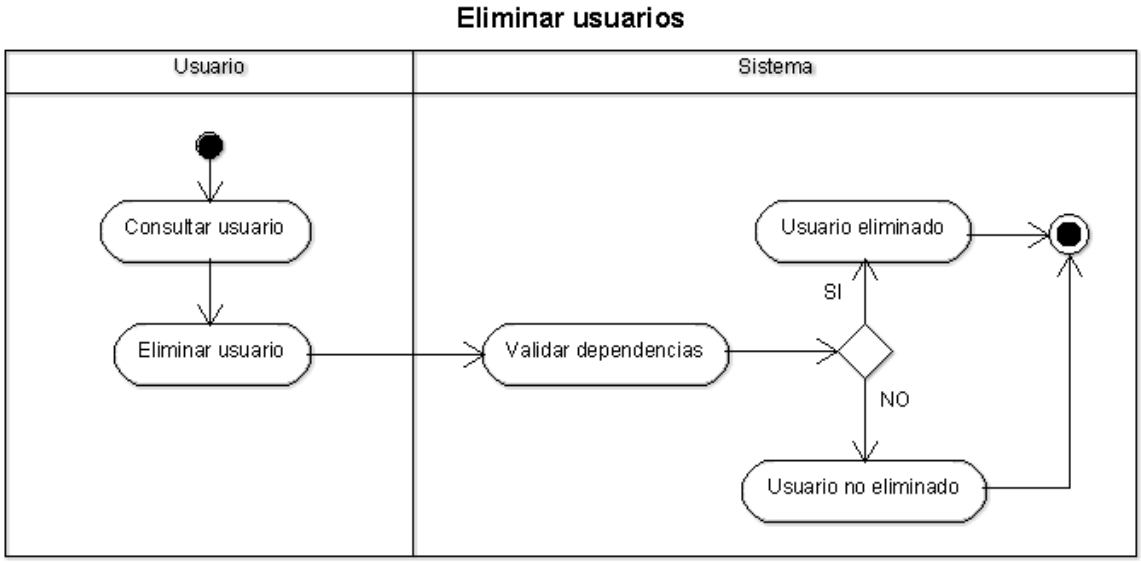


Figura 30. Actividad A10: Eliminar usuarios

9.2.11. ACTIVIDAD A11

La Figura 31 describe las actividades de un usuario cuando va a crear un impuesto en el sistema.

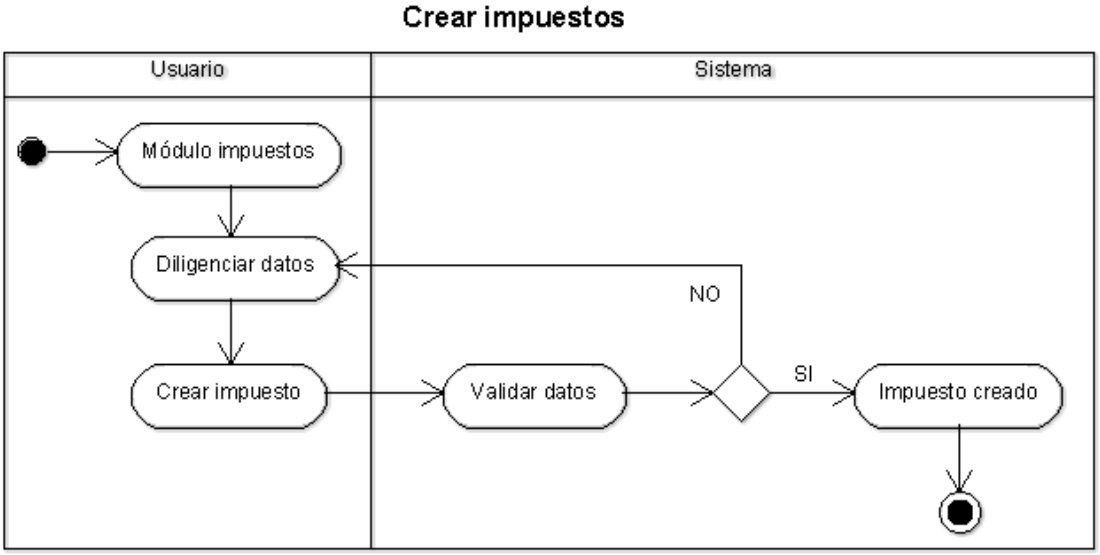


Figura 31. Actividad A11: Crear impuestos

9.2.12. ACTIVIDAD A12

La Figura 32 describe las actividades de un usuario cuando va a editar un impuesto en el sistema.

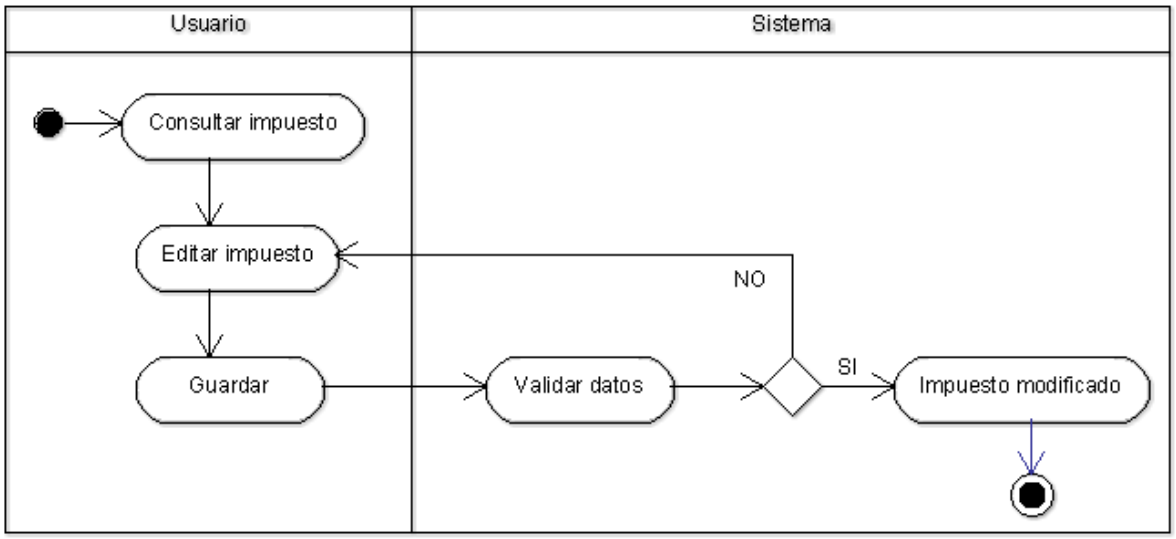


Figura 32. Actividad A12: Editar impuestos

9.2.13. ACTIVIDAD A13

La Figura 33 describe las actividades de un usuario cuando va a consultar un impuesto en el sistema.

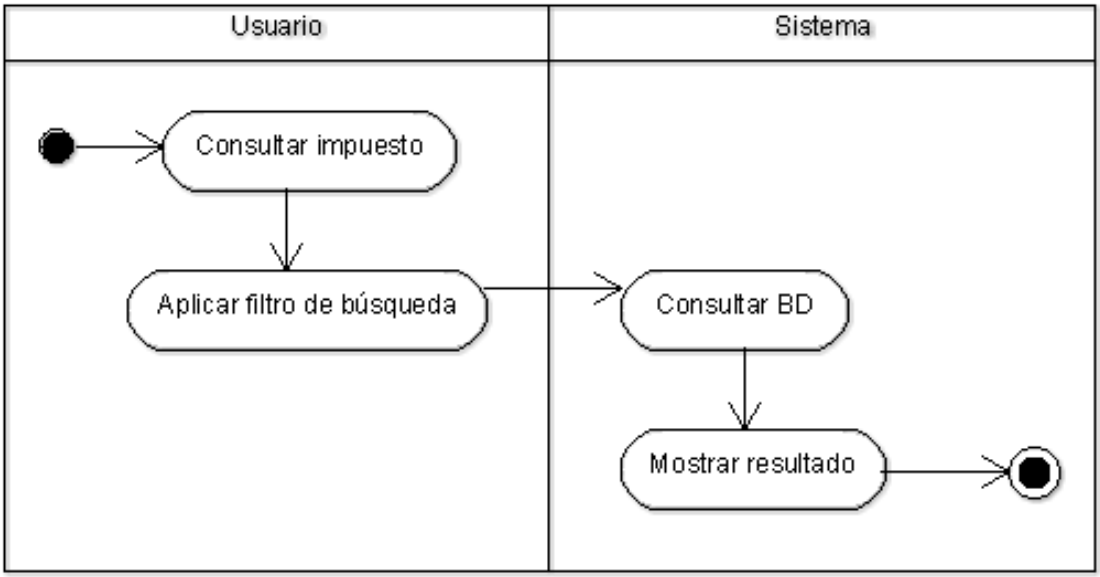


Figura 33. Actividad A13: Consultar impuestos

9.2.14. ACTIVIDAD A14

La Figura 34 describe las actividades de un usuario cuando va a eliminar un impuesto en el sistema.

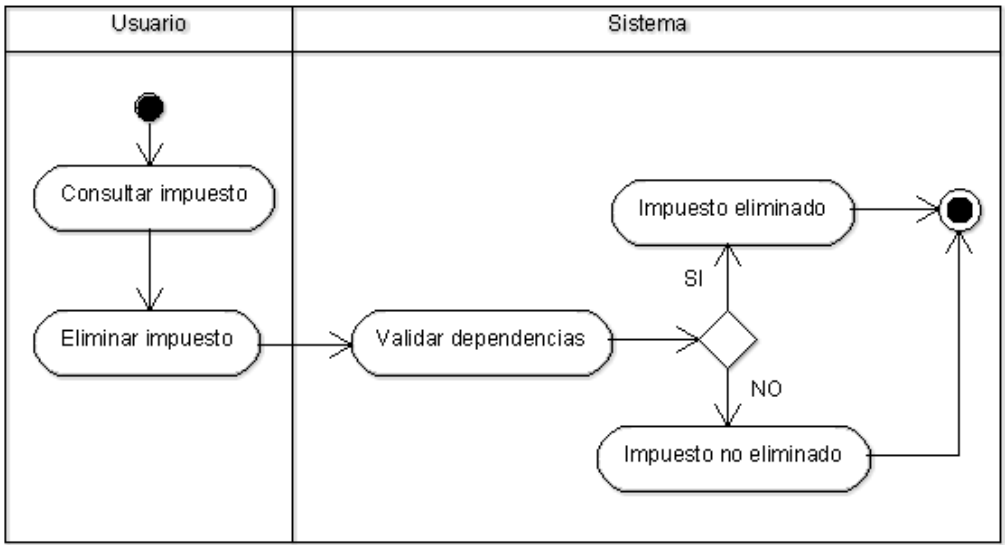


Figura 34. Actividad A14: Eliminar impuestos

9.3. DIAGRAMA DE BASE DE DATOS MONGODB

El sistema utiliza una base de datos no relacional MongoDB, ya que este tipo de base de datos permite realizar sistemas más abiertos y flexibles debido a su fácil adaptación de nuevas evoluciones de aplicaciones web

```
Account
{
    Name: String
    Lastname: String
    Company: String
    Country: String
    Email: String
    Status: Boolean
    Role: Number
    Password: String
    Creation: Date
    Idaccount: String
    Main: Number
}
```

```
Tax
{
    Country: String
    Wht: Number
}
```

```
Template
{
    name: String
    description: String
}
```

```
Pricing
{
    Title: String
    Company: String
    Idtemplate: Entero
    Región: String
    Idaccount: String
    Ec2instances: Array
    EbsStorage: Array
    storage: Array
    Rdsinstances: Array
    Creation: Date

}
```

Figura 35. Diagrama de base de datos NoSQL

9.4. ARQUITECTURA

De acuerdo con la Tabla 32, el sistema debe contar con un ambiente de producción altamente disponible, confiable y escalable, para lograrlo, se elige utilizar infraestructura sobre la nube pública de AWS. A continuación, se describen en detalle la arquitectura y características del ambiente final (producción). Ver Figura 36.

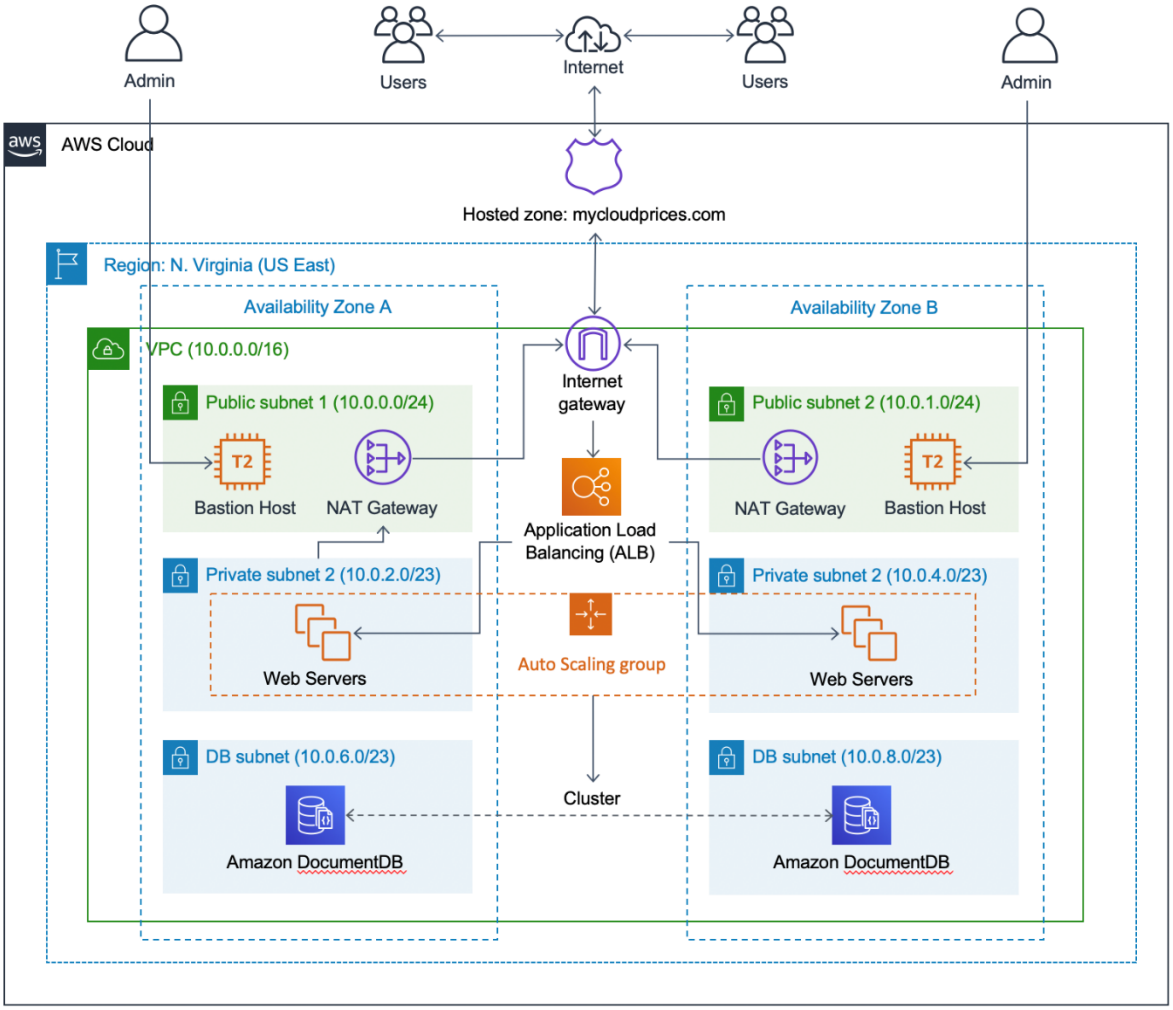


Figura 36. Arquitectura del sistema en Amazon Web Services.

La infraestructura AWS estará compuesta por:

- **Región N. Virginia:** Es un área geográficamente independiente de AWS que está compuesto por varios grupos de datacenter llamados zonas de disponibilidad. Esta región se encuentra en Estados Unidos y ofrece los mejores precios y la menor latencia en Colombia.

- **Zonas de disponibilidad:** Dos zonas de disponibilidad en la región de N. Virginia con el fin de garantizar alta disponibilidad del servicio en caso de desastres.

- **Red virtual:** Una red virtual privada que permite elegir el rango de direcciones ip que tendrá la infraestructura. Permite también dividir los recursos de cómputo en diferentes subredes dependiendo si los recursos necesitan acceso a internet (subred pública) o no es necesario exponerlos a internet (subred privada).

- **Route 53:** Un servicio de gestión de nombres de dominio que permite acceder a la aplicación a través del dominio www.mycloudprices.com.

- **Bastion Host:** Dos servidores virtuales Amazon EC2 linux desplegados cada uno en una zona de disponibilidad diferente cuyo propósito es brindar un acceso seguro únicamente al administrador de infraestructura para poder conectarse por SSH a los demás servidores de las capas inferiores (Servidores web y Servidores de Base de datos)

- **NAT Gateway:** Dos instancias NAT Gateway desplegadas cada uno en una zona de disponibilidad diferente para permitir a las instancias de la subred privada conectarse a internet.

- **Amazon Elastic Compute Cloud (EC2):** Dos servidores virtuales EC2 distribuidos en dos zonas de disponibilidad para garantizar una alta disponibilidad. En estos servidores estará instalado el código fuente del sistema.

- **Amazon EBS:** Volúmenes de almacenamiento persistente de bloques para utilizarlos como discos de storage o file system con las instancias de Amazon EC2. Se emplearán discos de estado sólido SSD para lograr una baja latencia y un desempeño constante de E/S de la aplicación.

- **Amazon Elastic Load Balancer (ELB):** Un balanceador de carga para la distribución de carga entre los servidores EC2 en las dos zonas de disponibilidad.

- **AWS Auto Scaling:** Un grupo de auto escalamiento que automatiza la manera en la que diferentes recursos responden ante los cambios que se producen en la demanda. Es decir, este servicio monitoriza la aplicación y añade o reduce automáticamente capacidad de computo en tiempo real de acuerdo con la demanda.

- **AWS Backup:** Un servicio automático para realizar las copias de respaldo de los servidores y bases de datos. Estas copias se almacenan en el servicio de Amazon S3 para garantizar durabilidad de la información respaldada.

- **Amazon DocumentDB:** Servicio de base de datos NoSQL compatible con MongodB donde se almacena los datos no estructurados de los usuarios y de la aplicación.

- **Amazon CloudWatch:** Permite monitorear todos los recursos de la infraestructura de AWS y de la aplicación. Amazon CloudWatch recopila y realiza seguimiento de métricas y logs, permite establecer alarmas y reaccionar automáticamente a los cambios en la infraestructura.

- **AWS CloudTrail:** Proporciona un historial de los eventos de la actividad de la cuenta de AWS para hacer seguimientos y auditorías operativas, relacionadas con las llamadas a las API en toda la infraestructura de AWS [30].

- **AWS Price List API:** Proporciona acceso a los precios de los servicios de AWS en forma JSON y CSV en tiempo real y actualizada.

La infraestructura de la Figura 36 es aprovisionada de forma automatizada por medio de código utilizando Terraform.

9.5. MOCKUPS

En esta sección se describen los mockups o maquetas digitales diseñados para visualizar las ideas y conceptos iniciales durante la fase de diseño del prototipo web:

Página web principal: Espacio público donde el usuario puede consultar las características del sistema, ver un video explicando cómo funciona el sistema y un formulario de contacto que puede utilizar para enviar comentarios, agradecimientos, quejas, reclamos, sugerencias, etc.

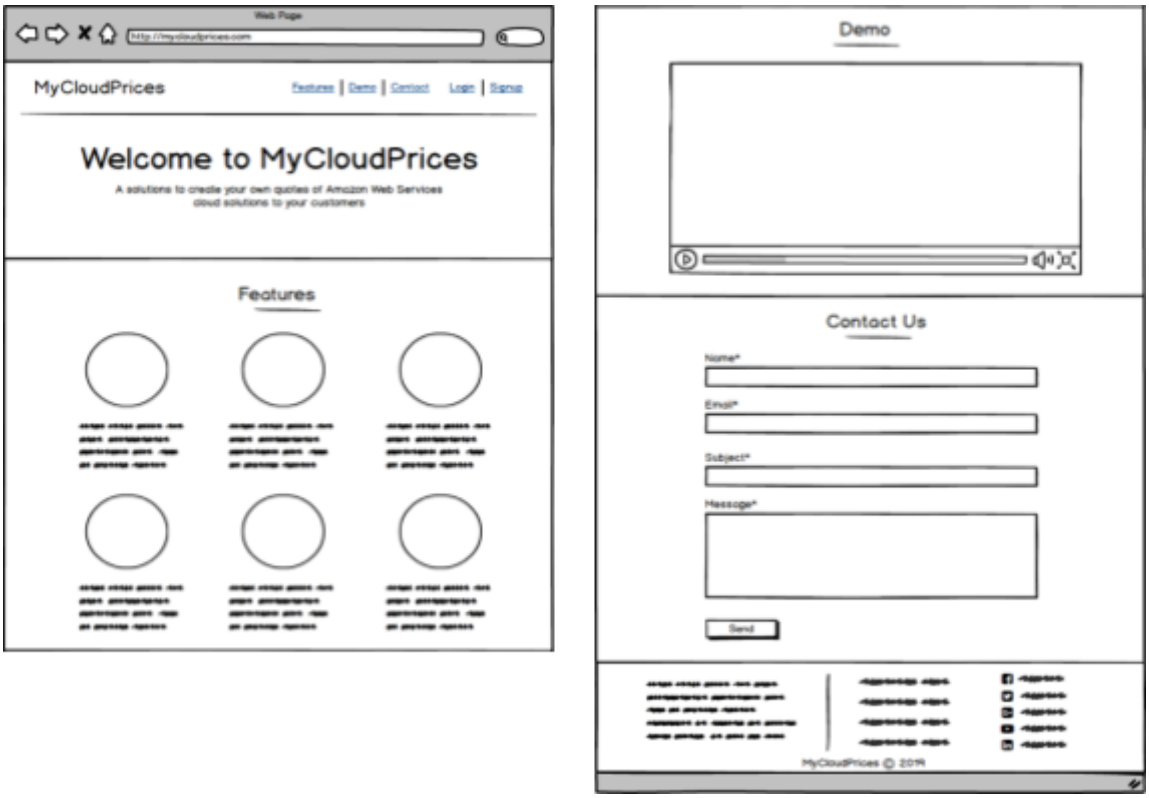


Figura 37. Mockup Página web principal

Crear cuenta: Módulo para la creación de una nueva cuenta. Cualquier usuario que consulte la página web puede crear una cuenta para tener acceso, esta cuenta inicialmente asume el rol de administrador que le garantiza acceso a todas las funcionalidades del sistema.

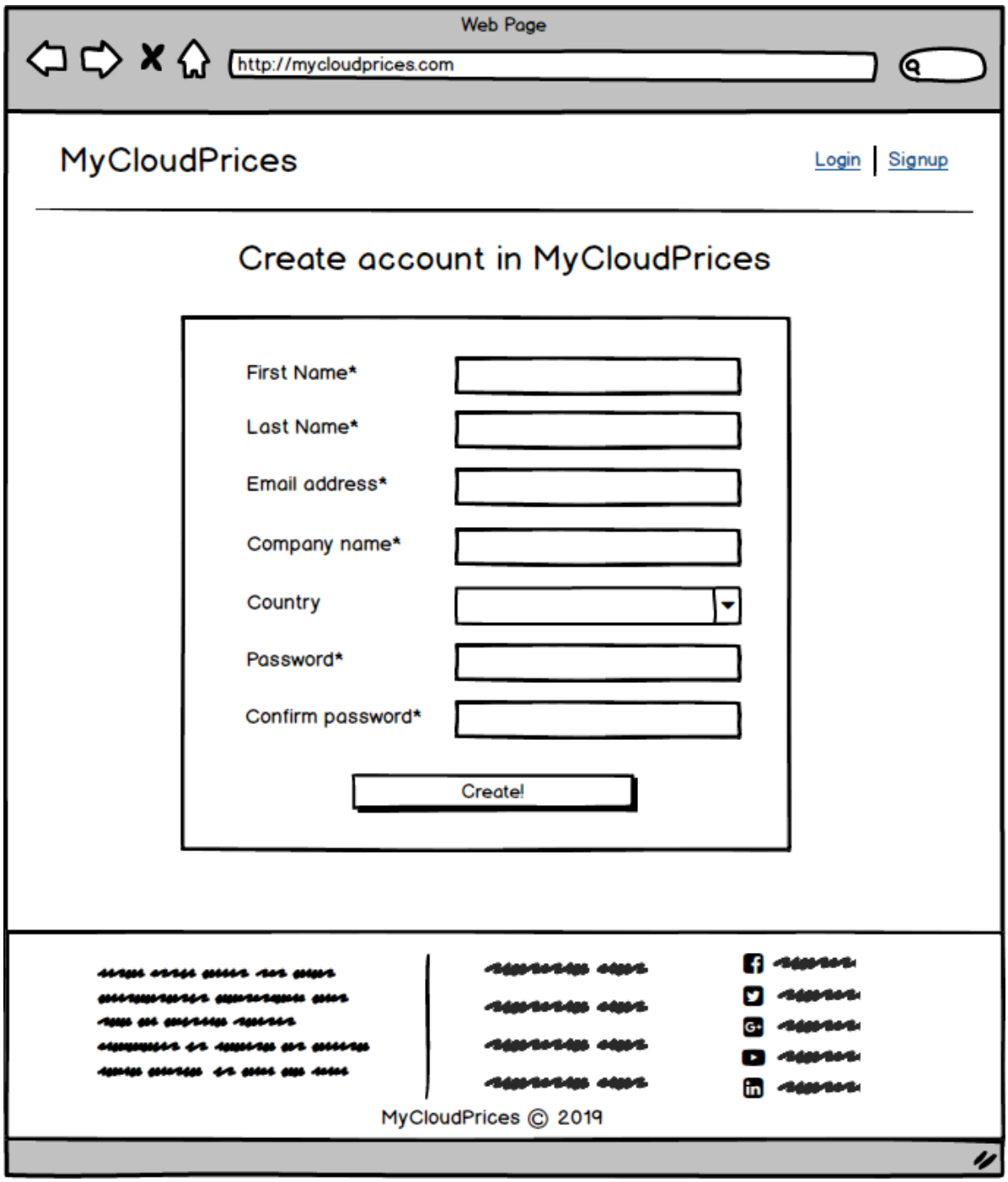


Figura 38. Mockup Crear cuenta

Autenticación: Formulario para la autenticación de usuarios registrados, tiene la opción de recuperar la contraseña en caso de olvidarla.

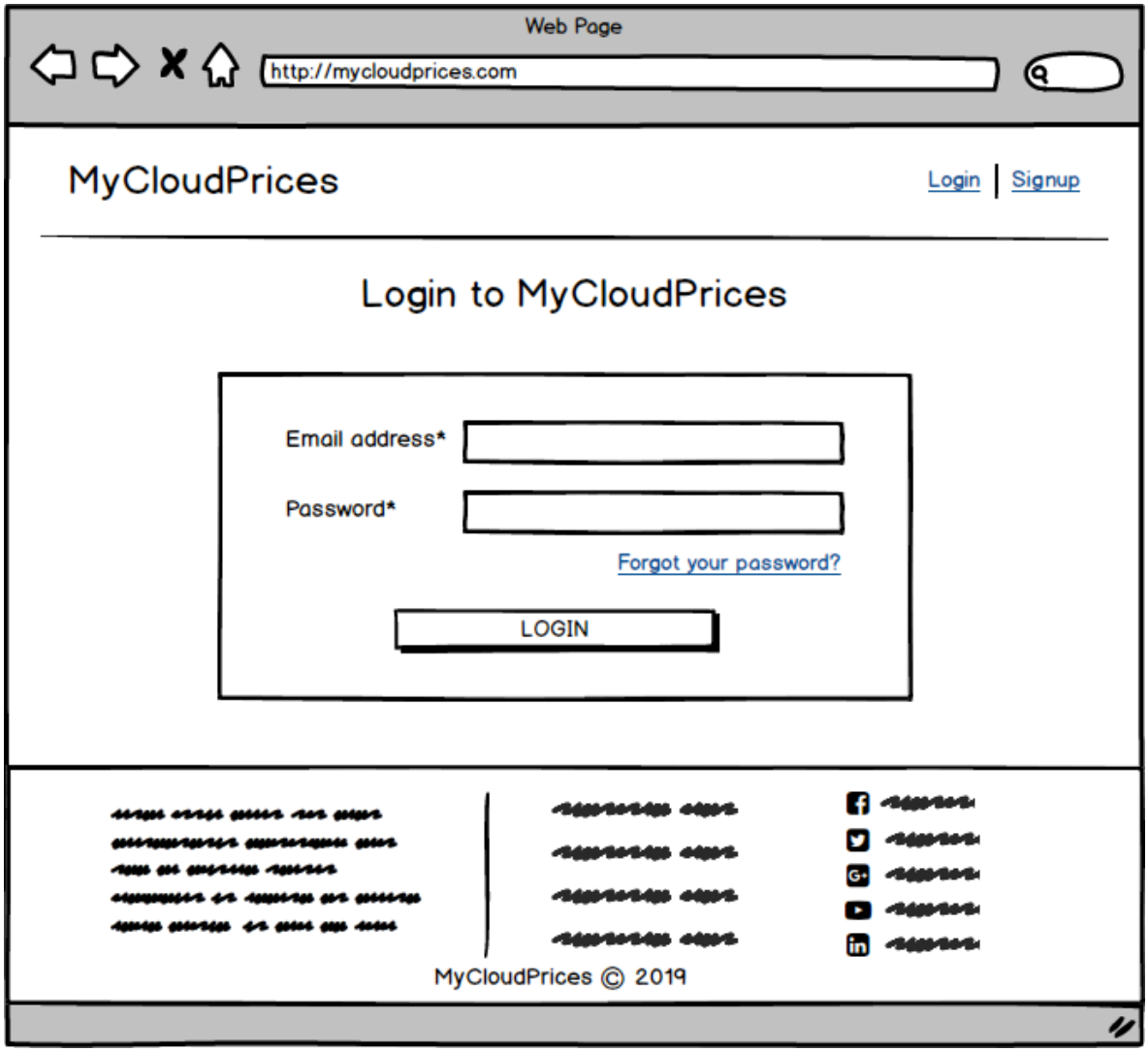


Figura 39. Mockup Autenticación

Mi cuenta: Funcionalidad para poder editar la información de la cuenta y cambiar la contraseña de acceso.

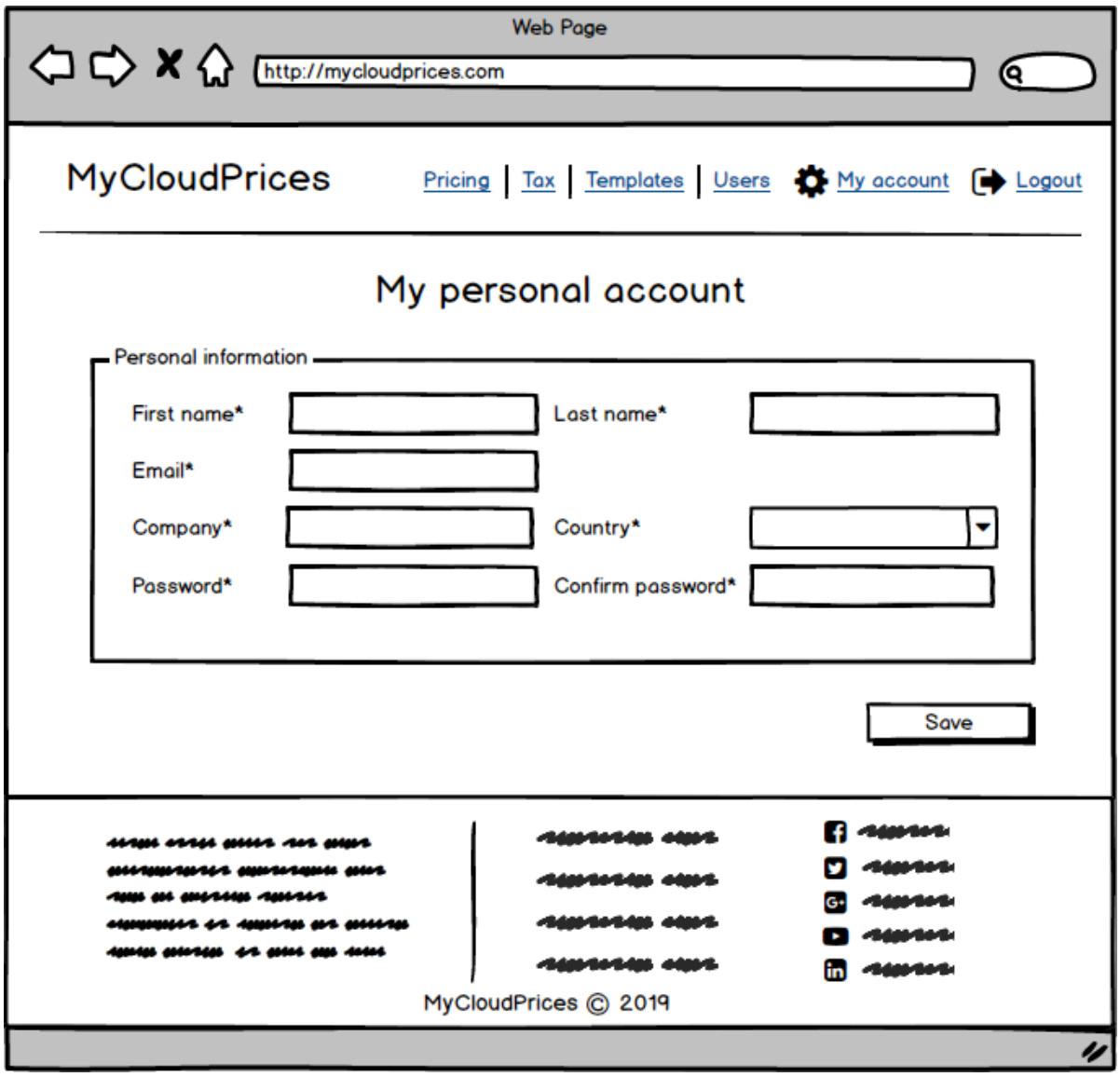


Figura 40. Mockup Mi cuenta

Impuestos: Módulo para la consulta y gestión de los porcentajes de los impuestos que se deben pagar por los servicios cloud dependiendo del país donde se vaya a realizar la facturación.

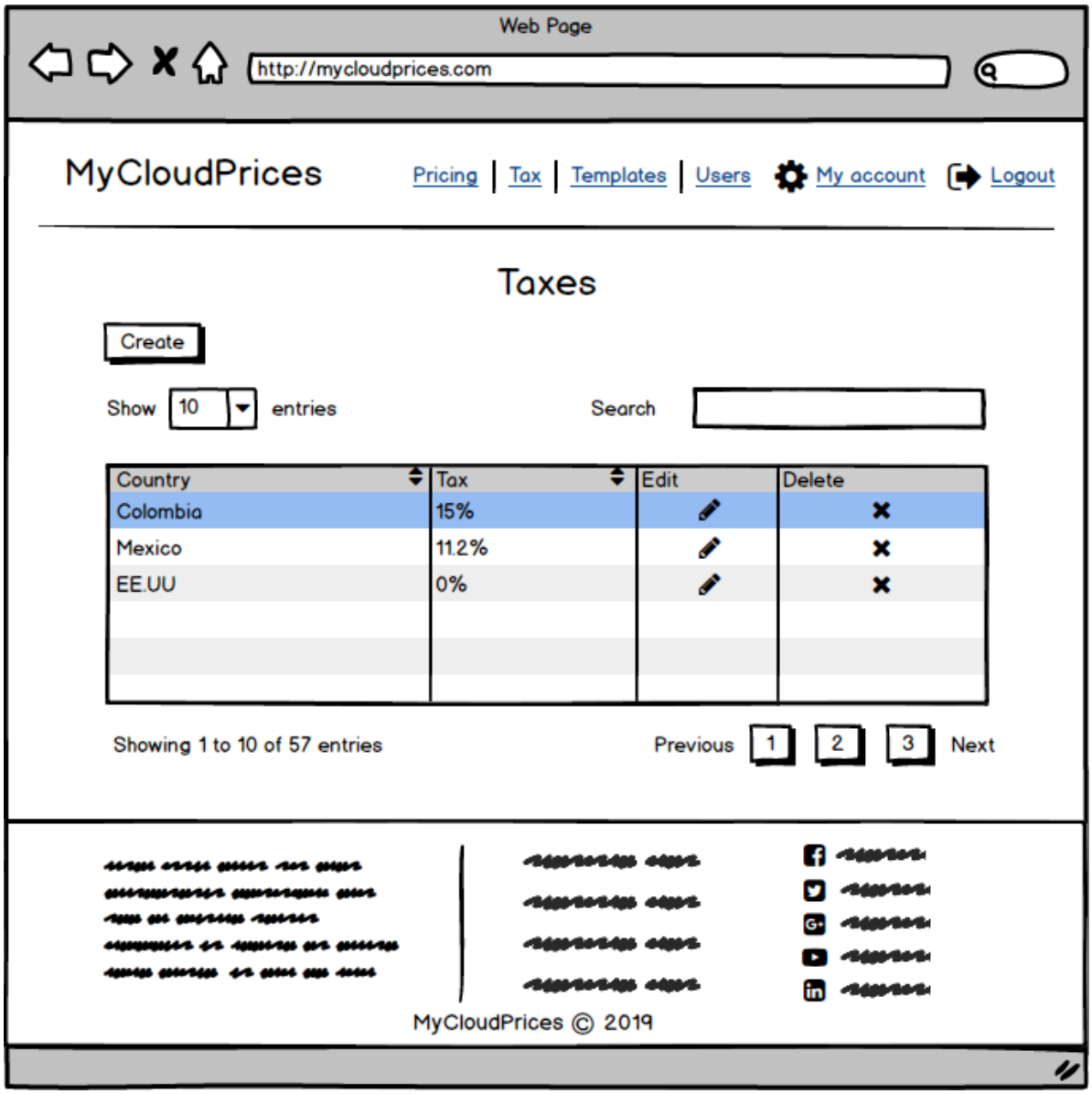


Figura 41. Mockup Impuestos

Crear impuestos: Formulario para la creación de los impuestos a aplicar por país a las cotizaciones que se generen.

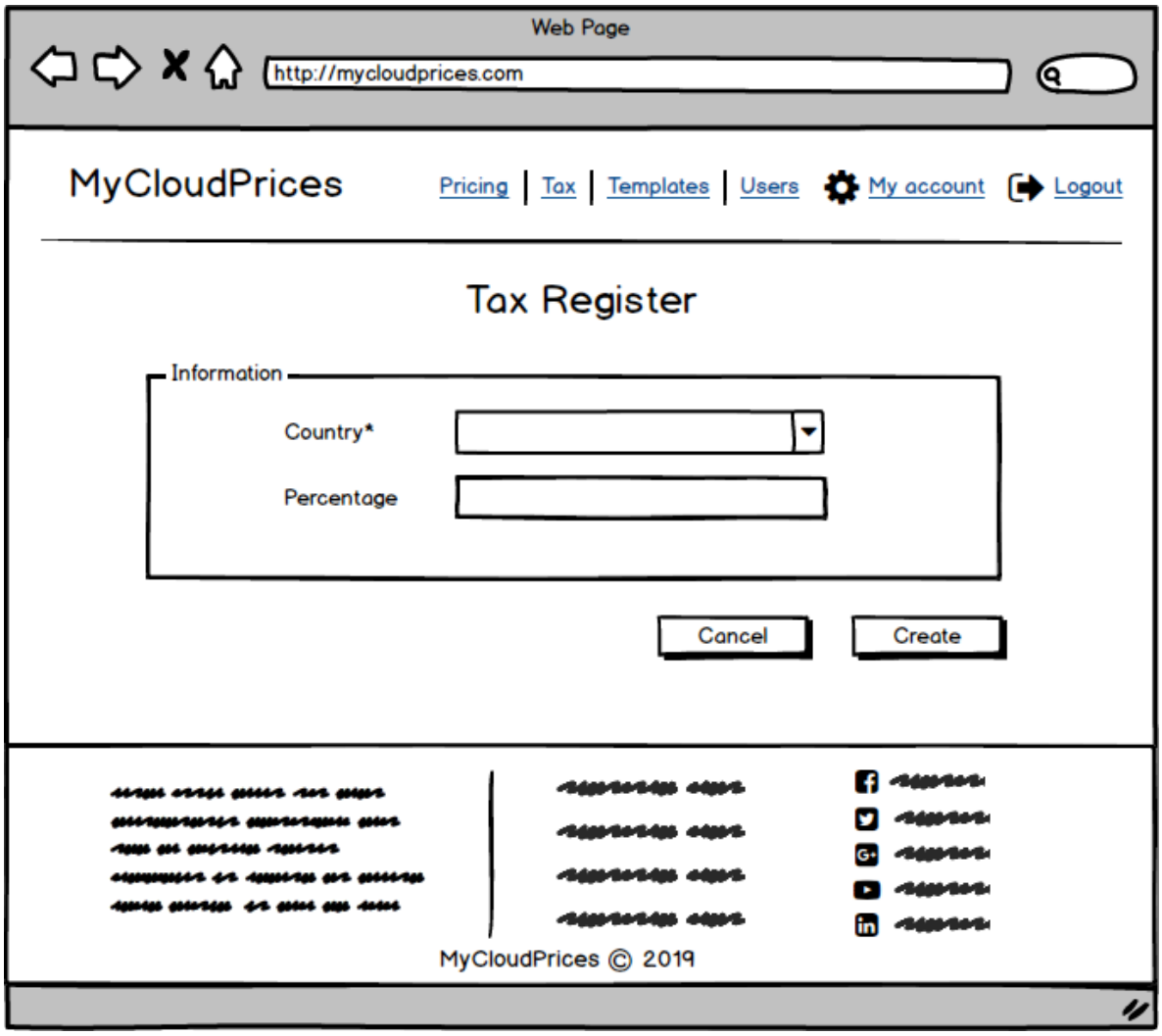


Figura 42. Mockup Crear impuestos

Editar impuestos: Formulario para la modificación de los impuestos.

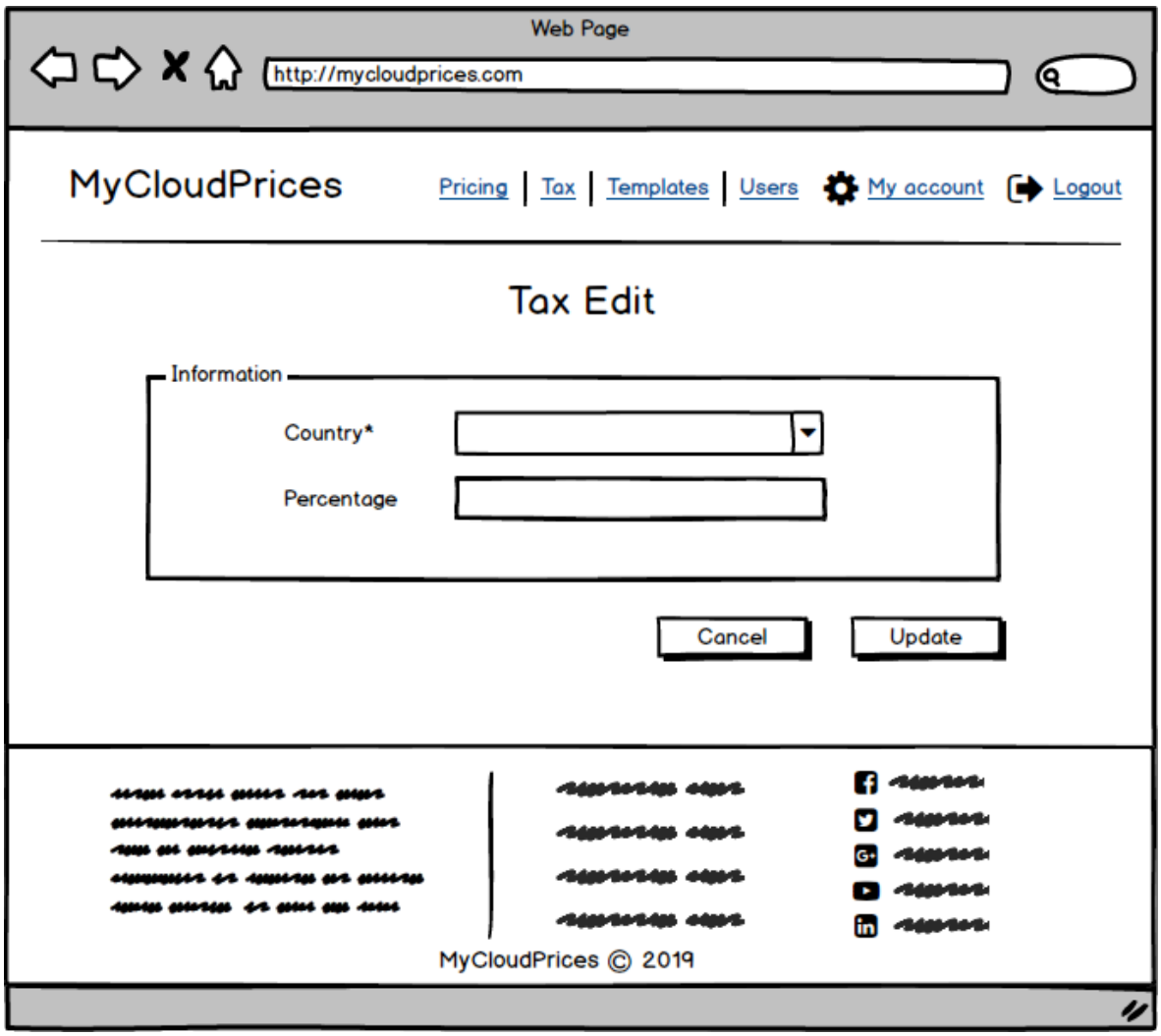


Figura 43. Mockup Editar impuestos

Módulo Cotizaciones: Módulo de consulta y gestión de cotizaciones registradas en el sistema.

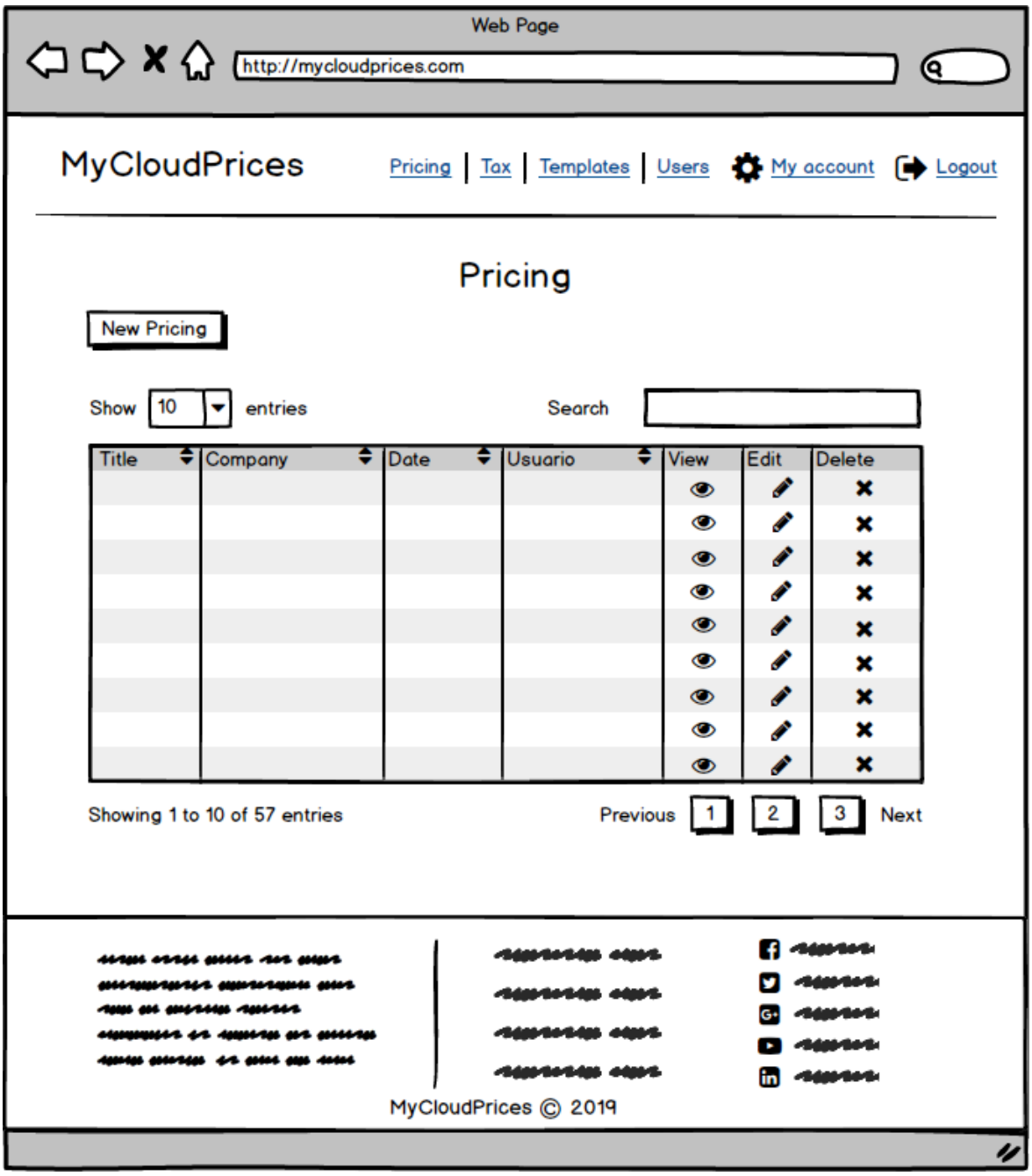


Figura 44. Mockup Módulo cotizaciones

Crear cotización: Formulario para la creación de cotizaciones de los servicios cloud de AWS.

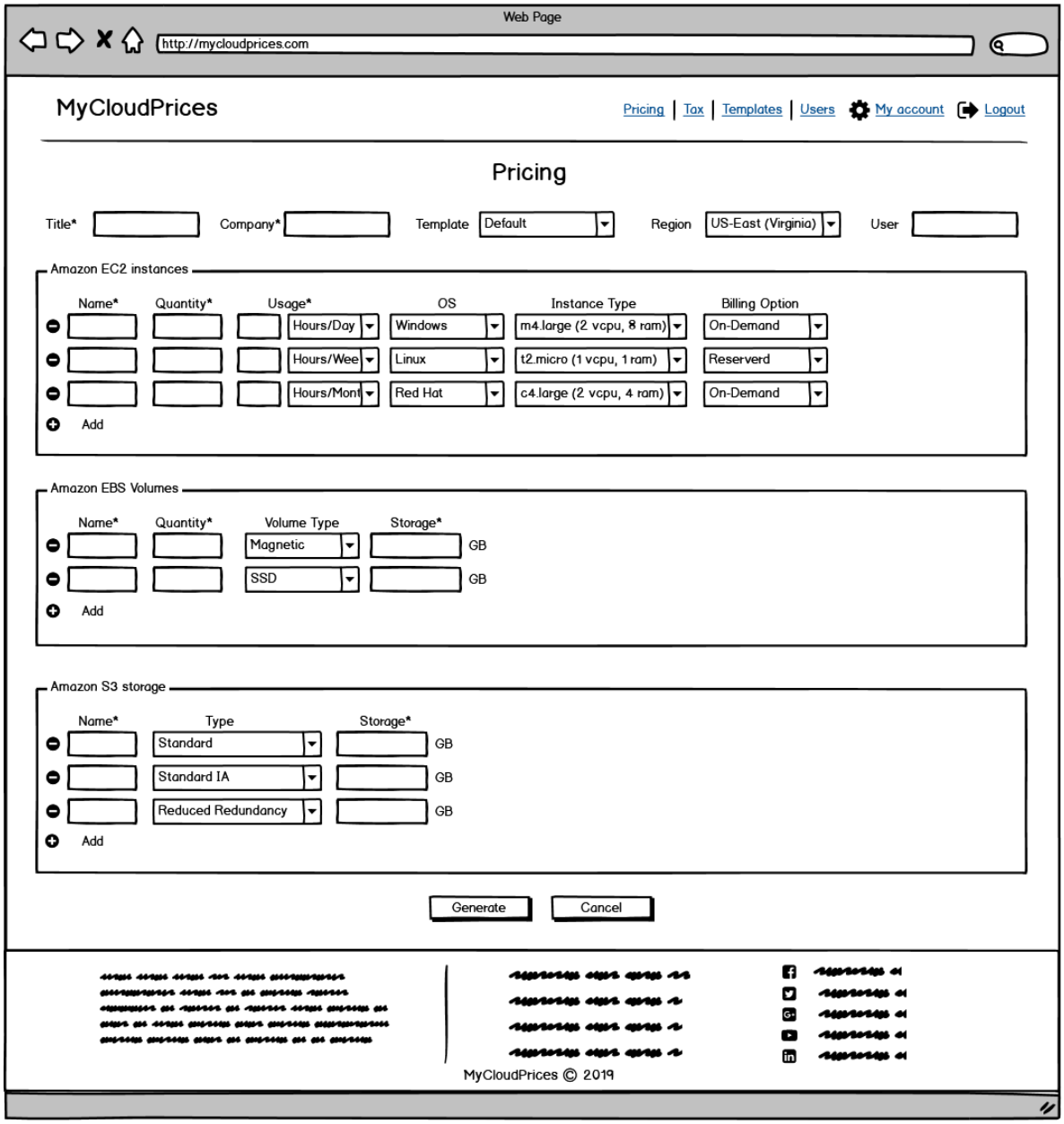


Figura 45. Mockup Crear cotización

Editar cotización: Formulario para la edición de cotizaciones de los servicios cloud de AWS.

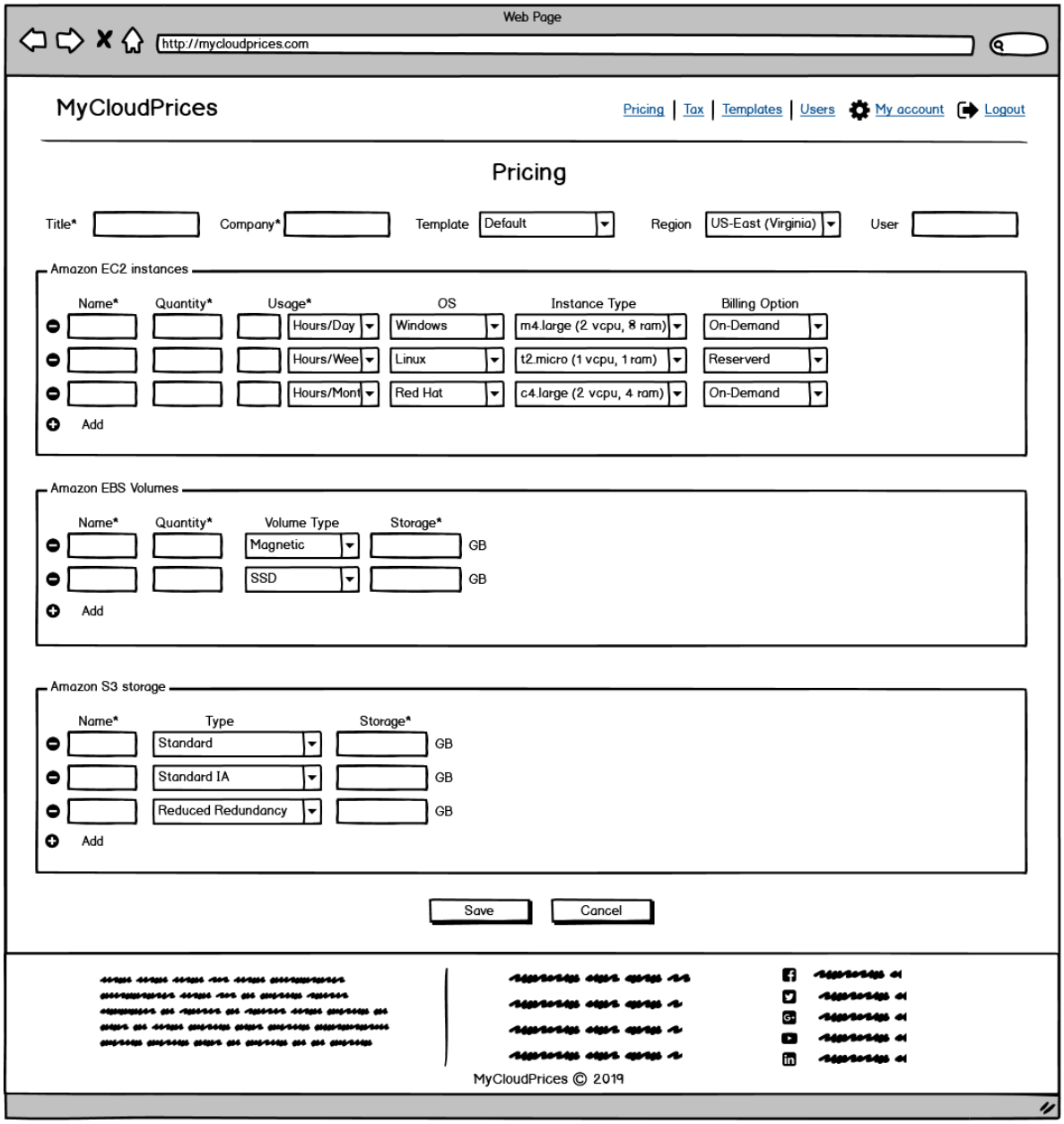


Figura 46. Mockup Editar cotización

Módulo Plantillas: Módulo para la visualización de plantillas disponibles en el sistema.

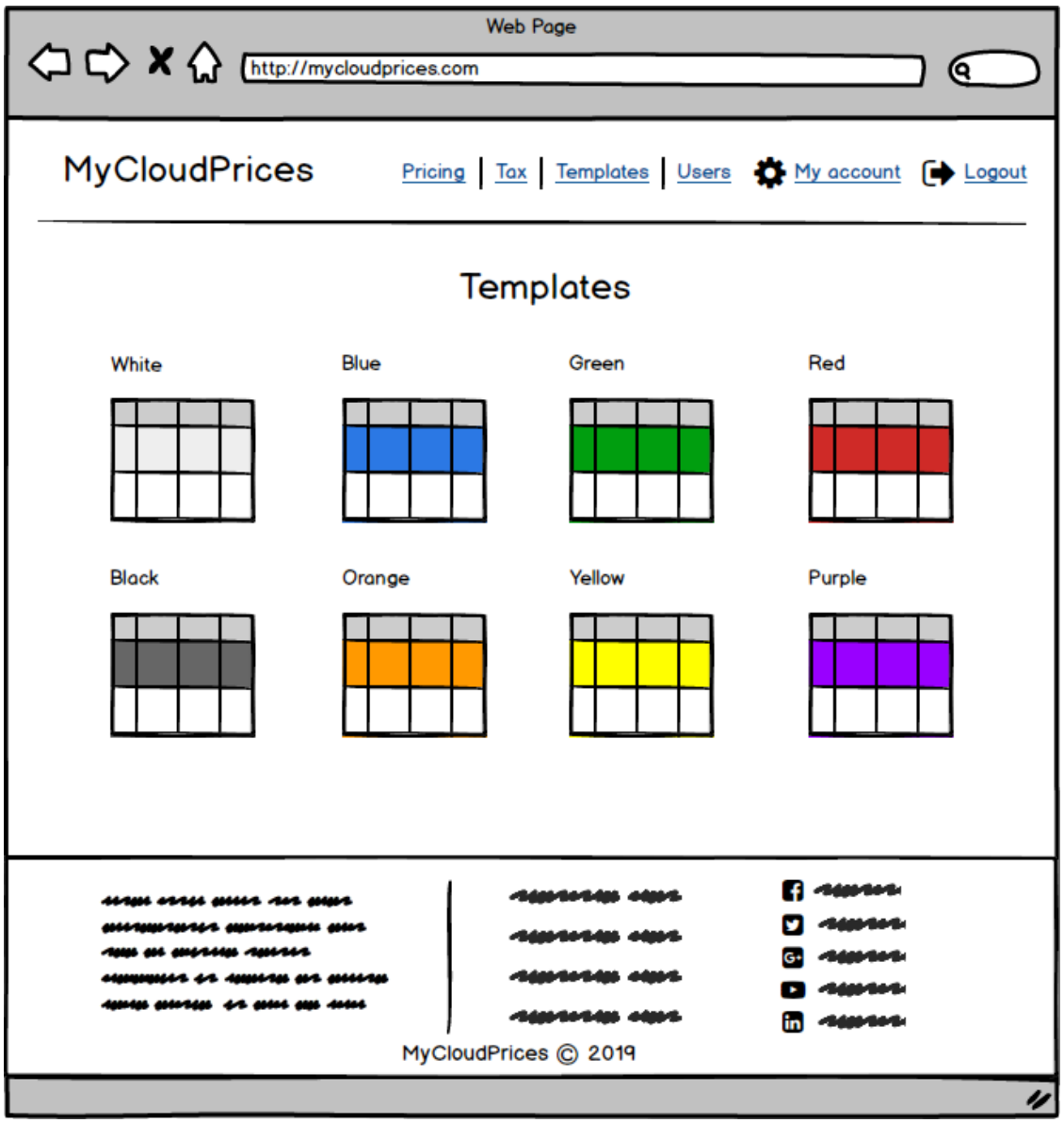


Figura 47. Mockup Módulo plantillas

Módulos usuarios: Módulo de gestión de usuarios asociados a la cuenta principal. Solo tiene acceso el administrador de la cuenta.

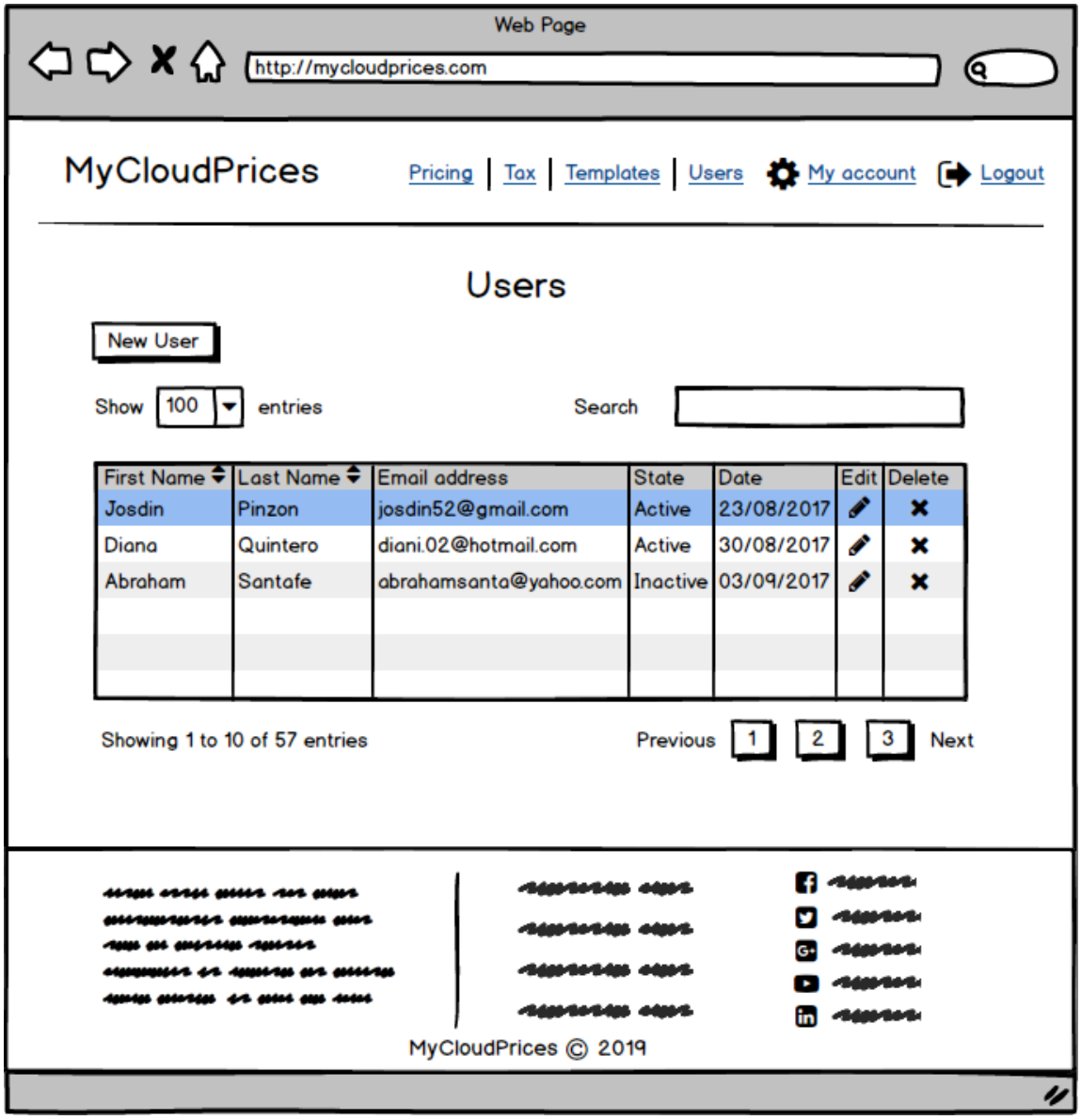


Figura 48. Mockup Módulo usuarios

Crear usuario: Formulario para la creación de usuarios.

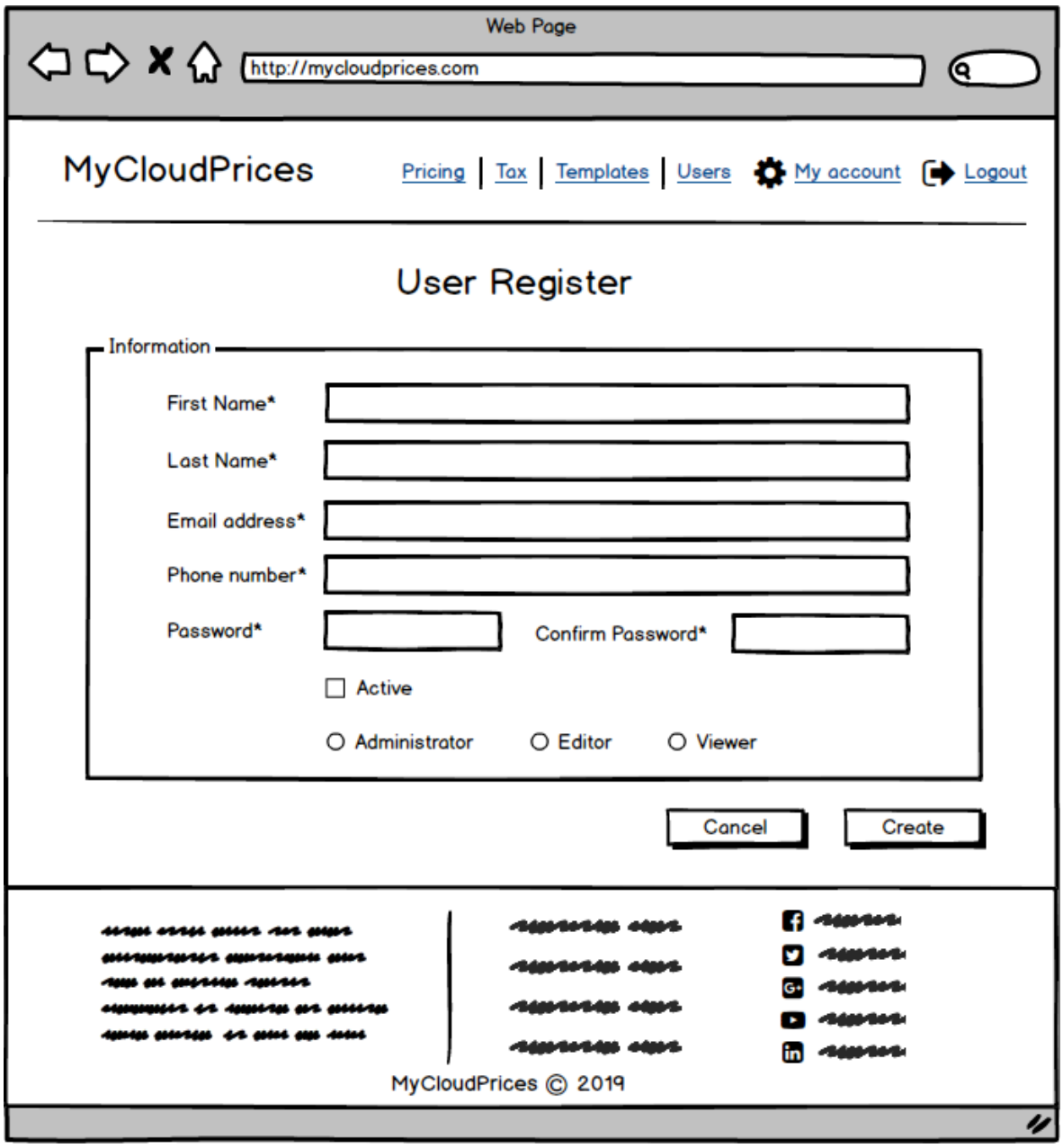


Figura 49. Mockup Crear usuario

Editar usuario: Formulario para la actualización de datos y permisos de los usuarios.

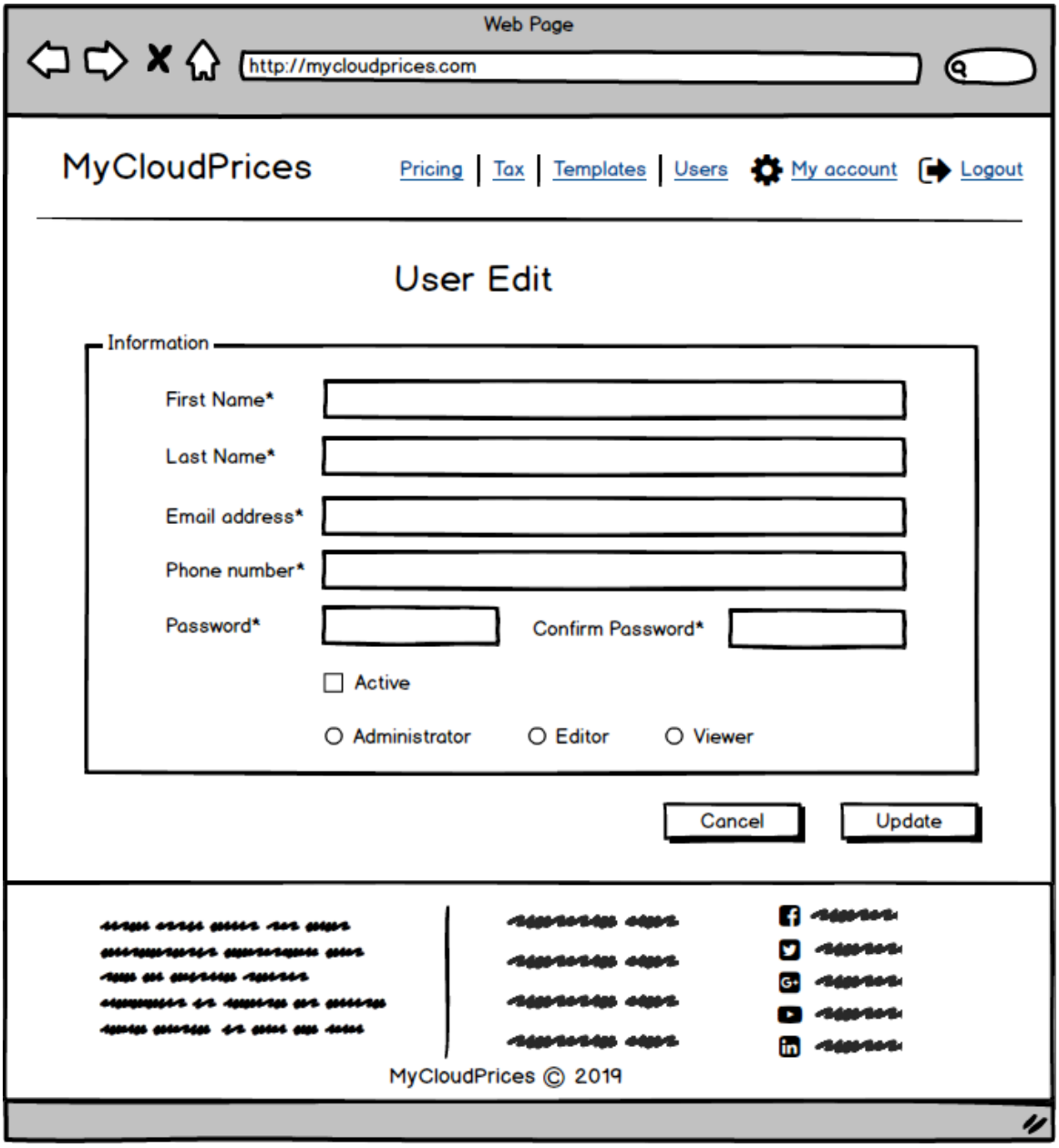


Figura 50. Mockup Editar usuario

10. FASE 3: DESARROLLO

En este capítulo se evidencia el desarrollo realizado del proyecto que incluye la descripción del ambiente de desarrollo, exposición del backlog priorizado y descripción de las actividades de cada sprint con sus respectivos resultados.

10.1. AMBIENTE DE DESARROLLO

En el ambiente de desarrollo se utiliza la computadora personal del autor MacBook Pro con sistema operativo macOS Movaje (Ver Figura 51). En ella se utiliza los siguientes componentes:

- IDE de desarrollo: **Visual Studio Code**
  Editor de código fuente creado por Microsoft totalmente gratuito y fácil de usar.

- Software de Control de Versiones: **Git**
  Integrado con Visual Studio Code permite ejecutar commits y push sin ejecutar ninguna línea de comando.

- Repositorio de código fuente: **GitHub**
  Servicio web que ofrece repositorios basados en Git públicos y privados.

- Servidor web: **Node.js**
  Entorno javascript del lado del servidor basado en eventos utilizado como servidor web del sistema

- Framework de desarrollo: **Express.js**
  Framework web para Node.js utilizado para programar la lógica del sistema.

- Base de datos no relacional: **MondoDB**
  Base de datos NoSQL orientado a documentos en formato JSON utilizado para almacenar la información del sistema

Figura 51. Macbook Pro. Tomado de: https://www.apple.com

10.2. BACKLOG PRIORIZADO DEL PRODUCTO

A continuación, el backlog priorizado donde se ordena las historias de usuario de acuerdo con el nivel de importancia y se establece una fecha de entrega para cada uno.

ÉPICA	HISTORIA DE USUARIO	SPRINT	FECHA	RESPONSABLE
E8	H23: Infraestructura	1	04/02/2019	Josdin Pinzón
E1	H1: Descripción del sistema de información	2	11/02/2019	Josdin Pinzón
E1	H2: Características del sistema de información	2	11/02/2019	Josdin Pinzón
E1	H4: Formulario de contacto	2	11/02/2019	Josdin Pinzón
E1	H5: Creación de cuenta	3	18/02/2019	Josdin Pinzón
E1	H6: Login de cuenta	4	25/02/2019	Josdin Pinzón
E1	H7: Configuración y preferencias de cuenta	4	25/02/2019	Josdin Pinzón
E5	H17: Listado de usuarios	5	04/03/2019	Josdin Pinzón
E5	H18: Crear usuarios	5	04/03/2019	Josdin Pinzón
E5	H19: Editar información usuario	6	11/03/2019	Josdin Pinzón
E5	H20: Eliminar usuarios	6	11/03/2019	Josdin Pinzón
E4	H13: Listado de impuestos	7	18/03/2019	Josdin Pinzón
E4	H14: Crear impuesto	7	18/03/2019	Josdin Pinzón
E4	H15: Editar información impuesto	8	25/03/2019	Josdin Pinzón
E4	H16: Eliminar impuesto	8	25/03/2019	Josdin Pinzón
E2	H8: Listado de plantillas	9	01/04/2019	Josdin Pinzón
E3	H9: Listado de cotizaciones	10	08/04/2019	Josdin Pinzón
E3	H10: Crear cotización	10	08/04/2019	Josdin Pinzón
E3	H11: Editar información cotizaciones	11	15/04/2019	Josdin Pinzón
E3	H12: Eliminar cotizaciones	11	15/04/2019	Josdin Pinzón
E6	H21: Avisos al usuario	12	22/04/2019	Josdin Pinzón
E7	H22: Funcionalidad fácil y sencilla	12	22/04/2019	Josdin Pinzón
E1	H3: Video demo del sistema de información	12	22/04/2019	Josdin Pinzón

Tabla 34. Backlog priorizado del producto

10.3. SPRINTS

De acuerdo con la metodología Scrum, el objetivo es lograr un entregable o un avance del proyecto en cada iteración denominada Sprint. Así que, el desarrollo del prototipo web se ha divido en 12 sprints, es decir, 1 sprint por semana, eso significa que son 12 entregables o avances que se describen a continuación.

10.3.1. SPRINT #1

El sprint #1 contiene las siguientes historias de usuario:

ÉPICA	HISTORIA DE USUARIO	SPRINT	Fecha	RESPONSABLE
E8	H23: Infraestructura	1	04/02/2019	Josdin Pinzón

Tabla 35. Sprint #1

El objetivo del Sprint #1 es realizar la historia de usuario H23 (Ver Tabla 32), que consiste en aprovisionar y automatizar el despliegue de la infraestructura AWS que soportará el prototipo web, para esto se genera un script llamado *infraestructure.tf* (Ver Anexo A) en el que contiene toda la configuración del despliegue de la infraestructura en AWS, esta configuración depende de unas variables que se han establecido en otros dos archivos: *variables.tf* (Ver Anexo B) y *terraform.tfvars* (Ver Anexo C).

10.3.2. SPRINT #2

El sprint #2 abarca las siguientes historias de usuario:

ÉPICA	HISTORIA DE USUARIO	SPRINT	Fecha	RESPONSABLE
E1	H1: Descripción del sistema de información	2	11/02/2019	Josdin Pinzón
E1	H2: Características del sistema de información	2	11/02/2019	Josdin Pinzón
E1	H4: Formulario de contacto	2	11/02/2019	Josdin Pinzón

Tabla 36. Sprint #2

El objetivo del Sprint #2 es realizar las historias de usuario H1 (Ver Tabla 10), H2 (Ver Tabla 11) y H4 (Ver Tabla 13), que consiste en crear la página web principal del prototipo web que incluye la descripción del sistema, características y el formulario de contacto.

Se elabora un archivo HTML con sus respectivas hojas de estilo CSS y archivos javascript que se puede consultar en el código fuente del proyecto. A continuación, se muestra del resultado final:

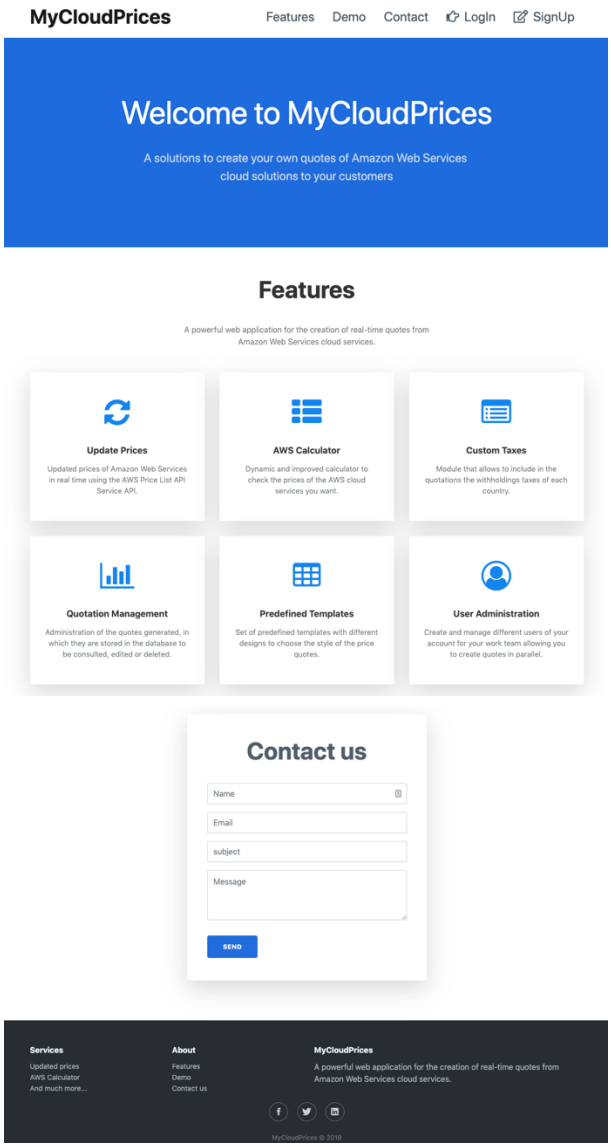


Figura 52. Página web MyCloudPrices

10.3.3. SPRINT #3

El sprint #3 abarca las siguientes actividades:

ÉPICA	HISTORIA DE USUARIO	SPRINT	Fecha	RESPONSABLE
E1	H5: Creación de cuenta	3	18/02/2019	Josdin Pinzón

Tabla 37. Sprint #3

El objetivo del Sprint #3 es crear la funcionalidad de creación de cuentas, por tanto, se crea el archivo HTML con su respectiva programación que se puede consultar en el código fuente del proyecto.

El código debe conectarse con la base de datos MongoDB para almacenar las cuentas creadas dentro del sistema. A continuación, se muestra el resultado obtenido.

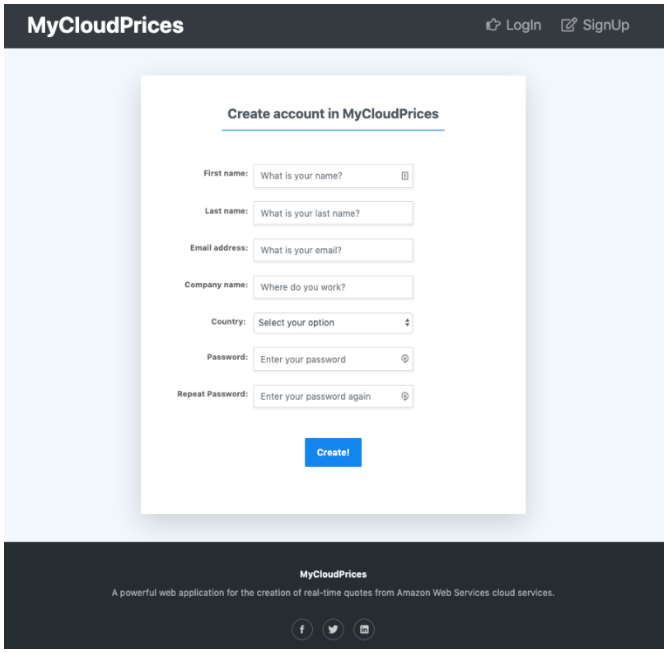


Figura 53. Crear cuenta en MyCloudPrices

10.3.4. SPRINT #4

El sprint #4 abarca las siguientes actividades:

ÉPICA	HISTORIA DE USUARIO	SPRINT	FECHA	RESPONSABLE
E1	H6: Login de cuenta	4	25/02/2019	Josdin Pinzón
E1	H7: Configuración y preferencias de cuenta	4	25/02/2019	Josdin Pinzón

Tabla 38. Sprint #4

El objetivo del Sprint #4 es crear la funcionalidad de autenticación e ingreso al sistema y el formulario para visualizar la información de la cuenta, para esto se elabora dos (2) archivos HTML con su respectiva programación que se puede consultar en el código fuente del proyecto.

El código debe conectarse con la base de datos MongoDB para consultar y comprobar que los datos ingresados pertenecen a una cuenta válida.

A continuación, se muestra el formulario de autenticación:

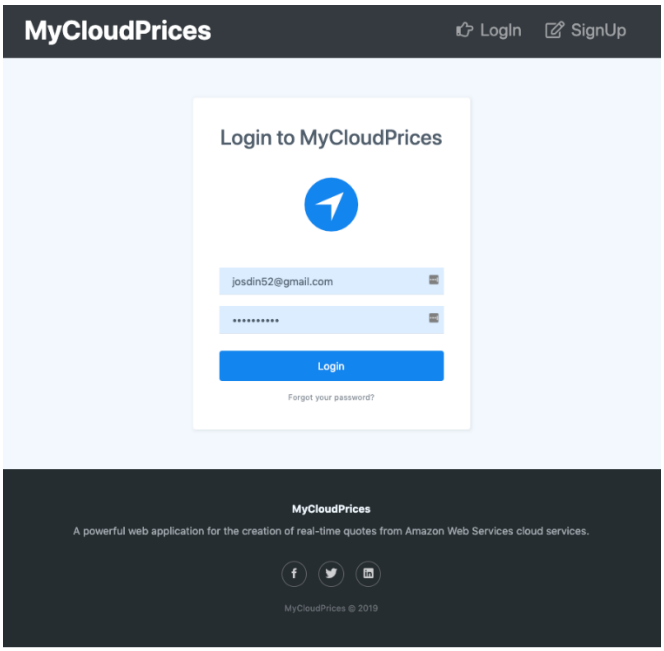


Figura 54. Login en MyCloudPrices

Al comprobar que los datos correo electrónico y contraseña son validos, el sistema se redirecciona a la sección de “Mi cuenta personal”, en donde el usuario tiene la posibilidad de editar su información personal y cambiar su contraseña.

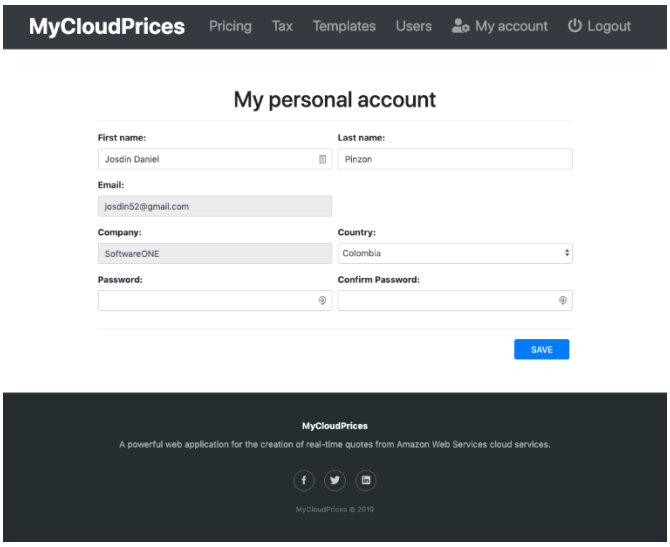


Figura 55. Mi cuenta en MyCloudPrices

10.3.5. SPRINT #5

El sprint #5 abarca las siguientes actividades:

ÉPICA	HISTORIA DE USUARIO	SPRINT	FECHA	RESPONSABLE
E5	H17: Listado de usuarios	5	04/03/2019	Josdin Pinzón
E5	H18: Crear usuarios	5	04/03/2019	Josdin Pinzón

Tabla 39. Sprint #5

El objetivo del Sprint #5 es crear el módulo de gestión de usuarios que incluye la consulta de los usuarios registrados bajo la cuenta principal y el formulario de creación de nuevos usuarios, los archivos de programación se pueden consultar en el código fuente del proyecto.

La tabla del listado de usuarios cuenta con:

- Filtro del número de registros que se desea visualizar.
- Campo de búsqueda para filtrar un usuario más rápidamente.
- Ordenamiento alfabético por cada columna.
- Panel de paginación.

A continuación, se muestra el listado de usuarios:

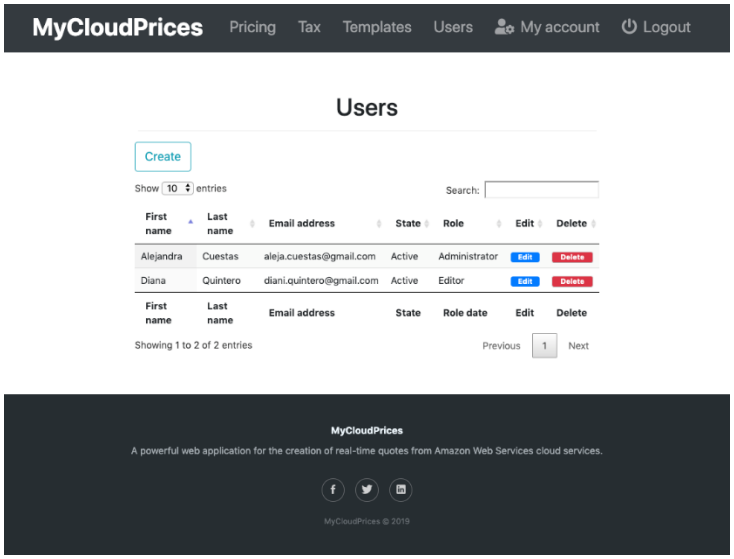


Figura 56. Listado de usuarios en MyCloudPrices

El formulario de creación de nuevos usuarios, válida que el correo electrónico no exista en la base de datos, de lo contrario el sistema no permite crear un usuario. A continuación, se muestra el formulario de creación de usuarios:

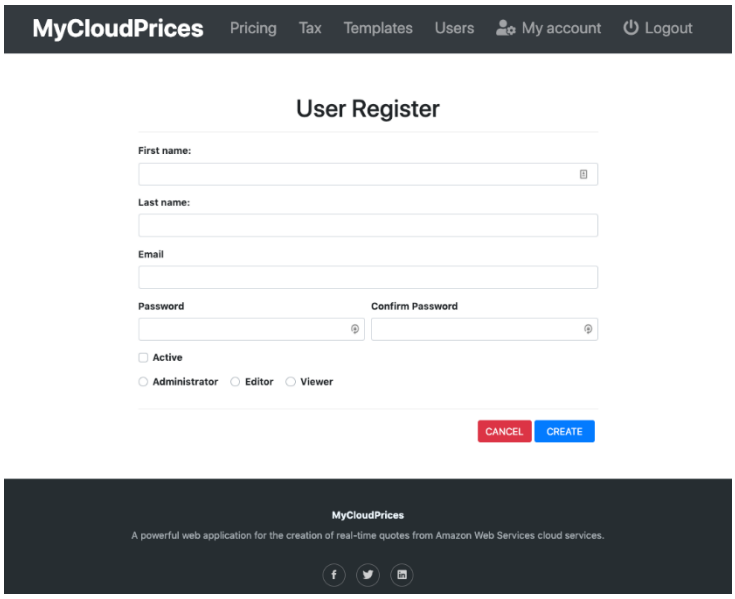


Figura 57. Crear usuario en MyCloudPrices

10.3.6. SPRINT #6

El sprint #6 abarca las siguientes actividades:

ÉPICA	HISTORIA DE USUARIO	SPRINT	FECHA	RESPONSABLE
E5	H19: Editar información usuario	6	11/03/2019	Josdin Pinzón
E5	H20: Eliminar usuarios	6	11/03/2019	Josdin Pinzón

Tabla 40. Sprint #6

El objetivo del Sprint #6 es crear la funcionalidad de editar y eliminar registros de usuarios, los archivos de programación se pueden consultar en el código fuente del proyecto.

A continuación, se muestra el formulario de edición de usuarios:

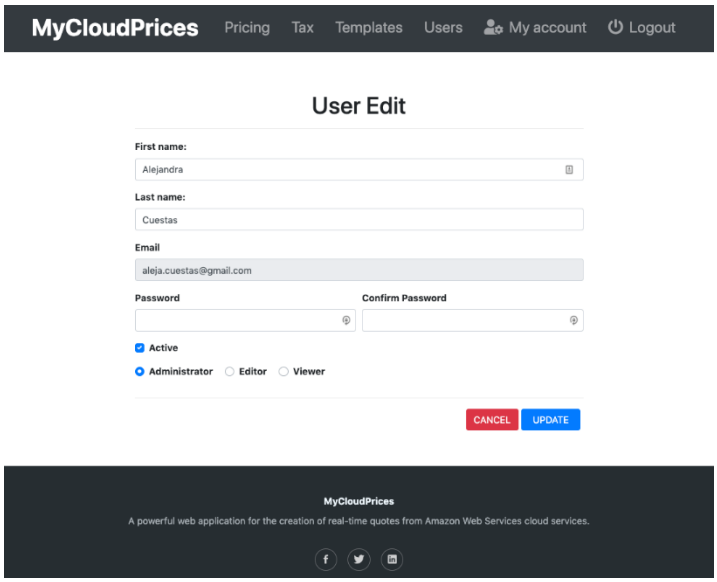


Figura 58. Editar usuario en MyCloudPrices

En la funcionalidad de eliminar, simplemente al hacer clic en el botón “Delete”, valida en la base de datos que el usuario no tenga cotizaciones creadas, de lo contrario no lo elimina. El sistema muestra un mensaje de confirmación.

10.3.7. SPRINT #7

El sprint #7 abarca las siguientes actividades:

ÉPICA	HISTORIA DE USUARIO	SPRINT	FECHA	RESPONSABLE
E4	H13: Listado de impuestos	7	18/03/2019	Josdin Pinzón
E4	H14: Crear impuesto	7	18/03/2019	Josdin Pinzón

Tabla 41. Sprint #7

El objetivo del Sprint #7 es crear el módulo de gestión de impuesto que incluye la consulta de los impuestos registrados y el formulario de creación de nuevos impuestos, los archivos de programación se pueden consultar en el código fuente del proyecto.

La tabla del listado de usuarios cuenta con:

- Filtro del número de registros que se desea visualizar.
- Campo de búsqueda para filtrar un impuesto más rápidamente.
- Ordenamiento alfabético por cada columna.
- Panel de paginación.

A continuación, se muestra el listado de usuarios:

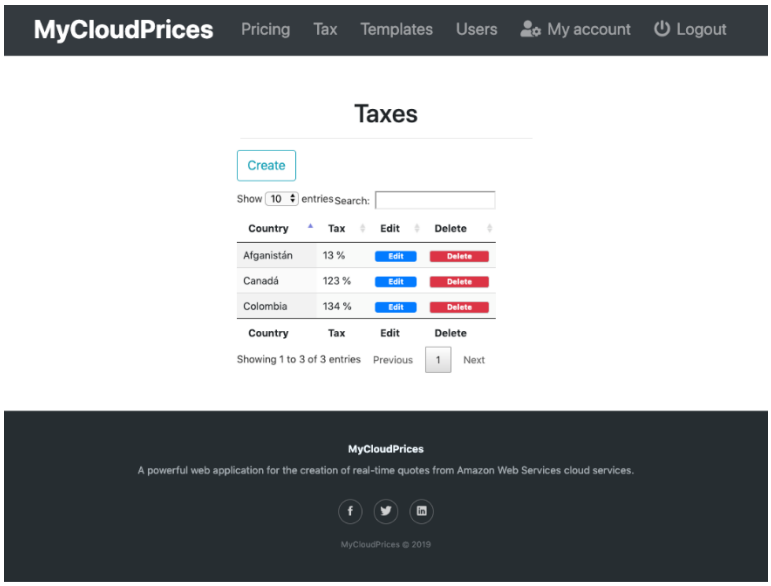


Figura 59. Listado de impuestos en MyCloudPrices

El formulario de creación de nuevos impuestos, válida que el país no exista en la base de datos, de lo contrario el sistema no permite crear un impuesto. A continuación, se muestra el formulario de creación de impuestos:

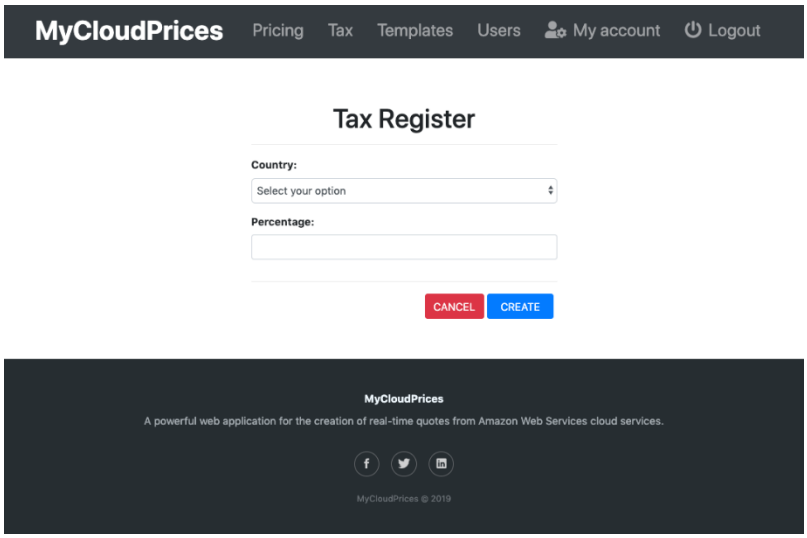


Figura 60. Crear impuesto en MyCloudPrices

10.3.8. SPRINT 8

El sprint #8 abarca las siguientes actividades:

ÉPICA	HISTORIA DE USUARIO	SPRINT	FECHA	RESPONSABLE
E4	H15: Editar información impuesto	8	25/03/2019	Josdin Pinzón
E4	H16: Eliminar impuesto	8	25/03/2019	Josdin Pinzón

Tabla 42. Sprint #8

El objetivo del Sprint #8 es crear la funcionalidad de editar y eliminar registros de impuestos, los archivos de programación se pueden consultar en el código fuente del proyecto.

A continuación, se muestra el formulario de edición de impuestos:

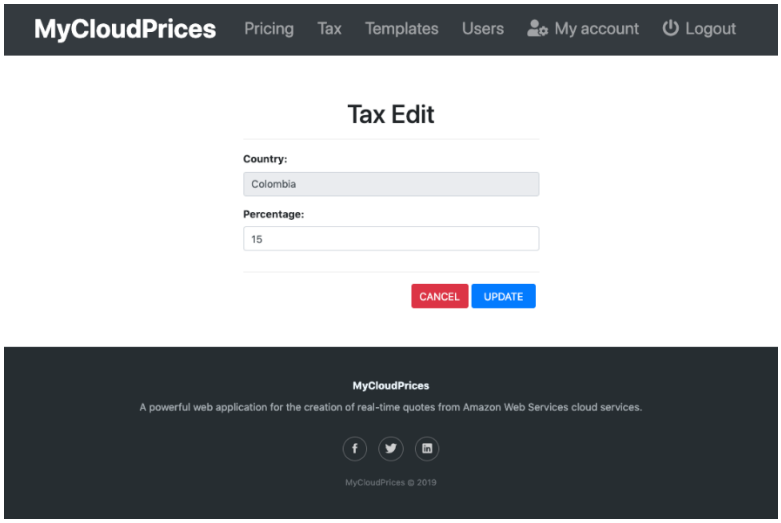


Figura 61. Editar impuesto en MyCloudPrices

En la funcionalidad de eliminar, simplemente al hacer clic en el botón “Delete”, valida en la base de datos que el impuesto no tenga cotizaciones asociadas, de lo contrario no lo elimina. El sistema muestra un mensaje de confirmación.

10.3.9. SPRINT 9

El sprint #9 abarca las siguientes actividades:

ÉPICA	HISTORIA DE USUARIO	SPRINT	FECHA	RESPONSABLE
E2	H8: Listado de plantillas	9	01/04/2019	Josdin Pinzón

Tabla 43. Sprint #9

El objetivo del Sprint #9 es crear el módulo de visualización y consulta de plantillas predeterminadas que cuenta el sistema para el estilo y diseño de la tabla de la cotización, los archivos de programación se pueden consultar en el código fuente del proyecto.

A continuación, se muestra el listado de las plantillas:

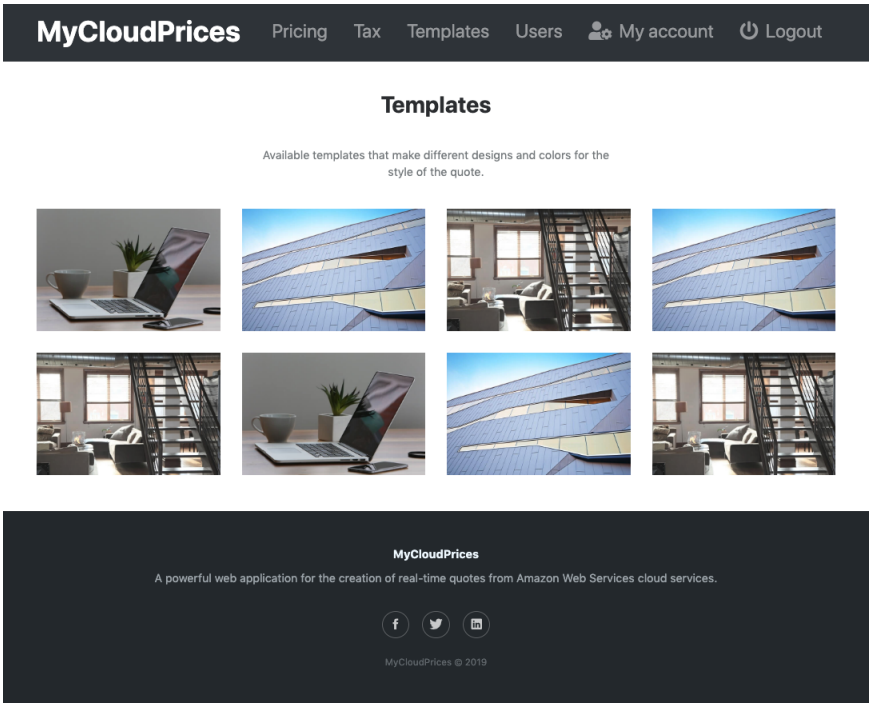


Figura 62. Plantillas en MyCloudPrices

11. FASE 4: IMPLEMENTACIÓN Y PRUEBAS

En este capítulo se evidencia el proceso de implementación del prototipo web en un ambiente productivo, pruebas unitarias a cada funcionalidad, pruebas de desempeño y pruebas de recuperación y tolerancia a fallos.

11.1. IMPLEMENTACIÓN

Para el aprovisionamiento de la infraestructura se utiliza el script elaborado en el Sprint #1 (Ver Anexo A
Infrastructure.tf). Para iniciar el proceso de aprovisionamiento se deben ejecutar los siguientes comandos:

terraform init

El resultado de este comando puede ser similar a esto:

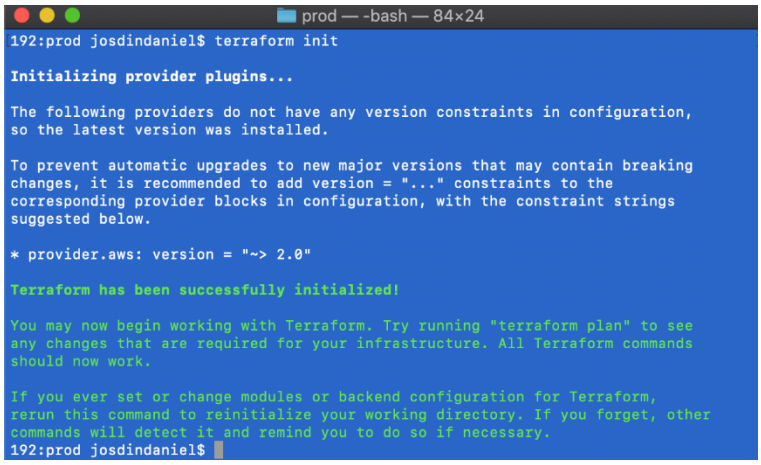


Figura 63. Iniciar Terraform

Este comando se utiliza para inicializar un directorio de trabajo que contiene los archivos de configuración de Terraform. Este es el primer comando que debe ejecutarse después de escribir una nueva configuración de Terraform.

Luego se ejecuta el siguiente comando:

terraform plan

El resultado de este comando puede ser similar a esto:

```
prod — -bash — 105×33

  + aws_vpc.vpc-mcp
      id:                                     <computed>
      arn:                                    <computed>
      assign_generated_ipv6_cidr_block:       "false"
      cidr_block:                             "10.0.0.0/16"
      default_network_acl_id:                 <computed>
      default_route_table_id:                 <computed>
      default_security_group_id:              <computed>
      dhcp_options_id:                        <computed>
      enable_classiclink:                     <computed>
      enable_classiclink_dns_support:         <computed>
      enable_dns_hostnames:                   "true"
      enable_dns_support:                     "true"
      instance_tenancy:                       "default"
      ipv6_association_id:                    <computed>
      ipv6_cidr_block:                        <computed>
      main_route_table_id:                    <computed>
      owner_id:                               <computed>
      tags.%:                                 "2"
      tags.Env:                               "prod"
      tags.Name:                              "vpc-mcp"


Plan: 40 to add, 0 to change, 0 to destroy.

------------------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.

192:prod josdindaniel$
```

Figura 64. Terraform Plan

Este comando se utiliza para crear un plan de ejecución. Terraform realiza una actualización, y luego determina qué acciones son necesarias para lograr el estado deseado especificado en los archivos de configuración. Como se puede observar en la Figura 64 se informa que se van a ejecutar 40 configuraciones para el aprovisionamiento de la infraestructura en AWS.

Para aceptar la configuración se ejecuta el siguiente comando:

terraform apply

Terraform empieza a ejecutar el aprovisionamiento y ejecución de la configuración de infraestructura. Este proceso puede demorarse aproximadamente 10 minutos.

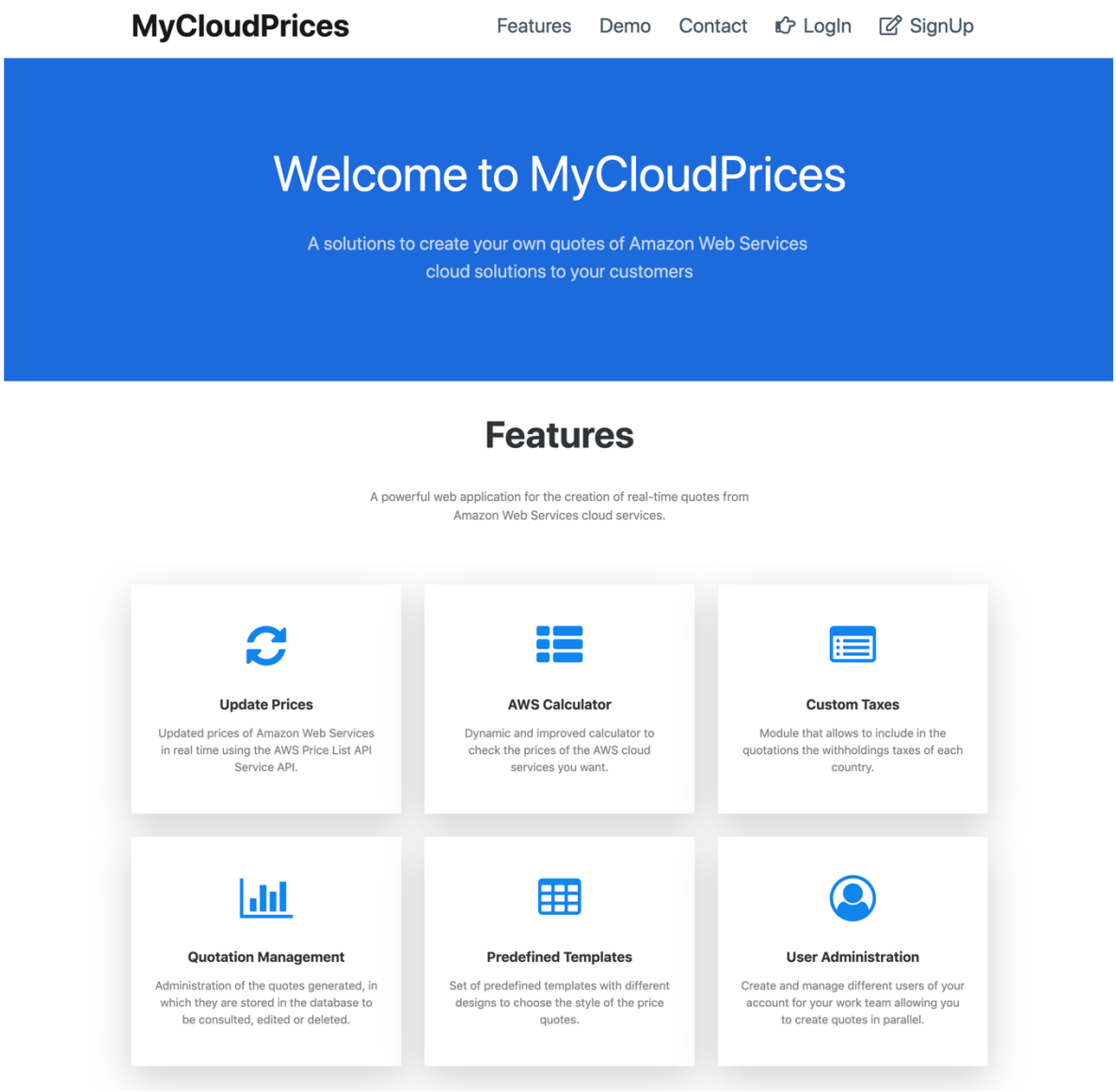
Finalmente, en el navegador, ingresamos a la url www.mycloudprices.com y se puede visualizar la página web principal del prototipo web.

11.2. PRUEBAS

Una vez realizada la implementación se procede a realizar pruebas unitarias del funcionamiento del sistema, en otras palabras, se probará cada una de las historias de usuario descritas en el punto 8.3 del presente documento.

11.2.1. PRUEBAS FUNCIONALES

- Navegación página web



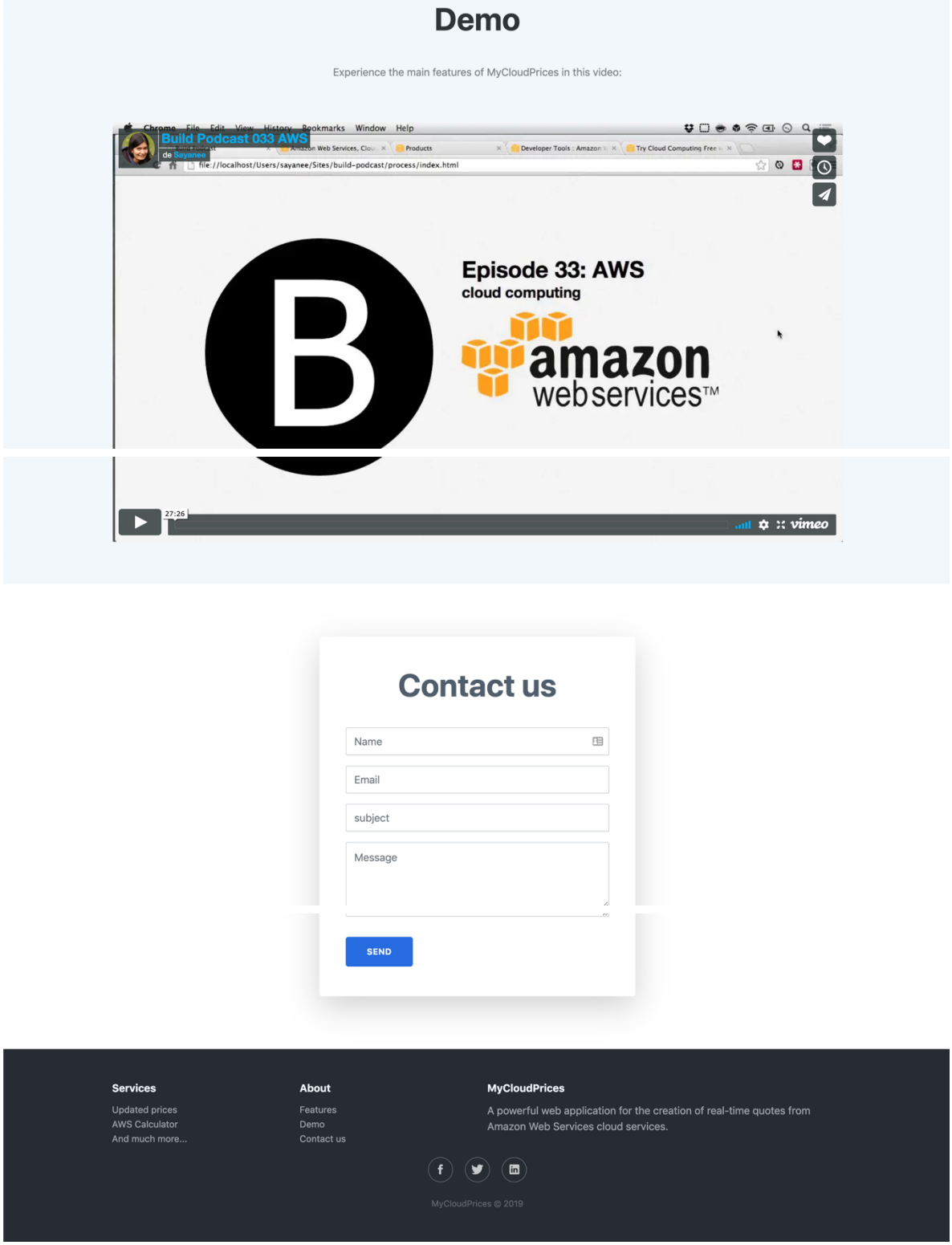


Figura 65. Prueba: Navegación web

Eda

- Creación de una cuenta

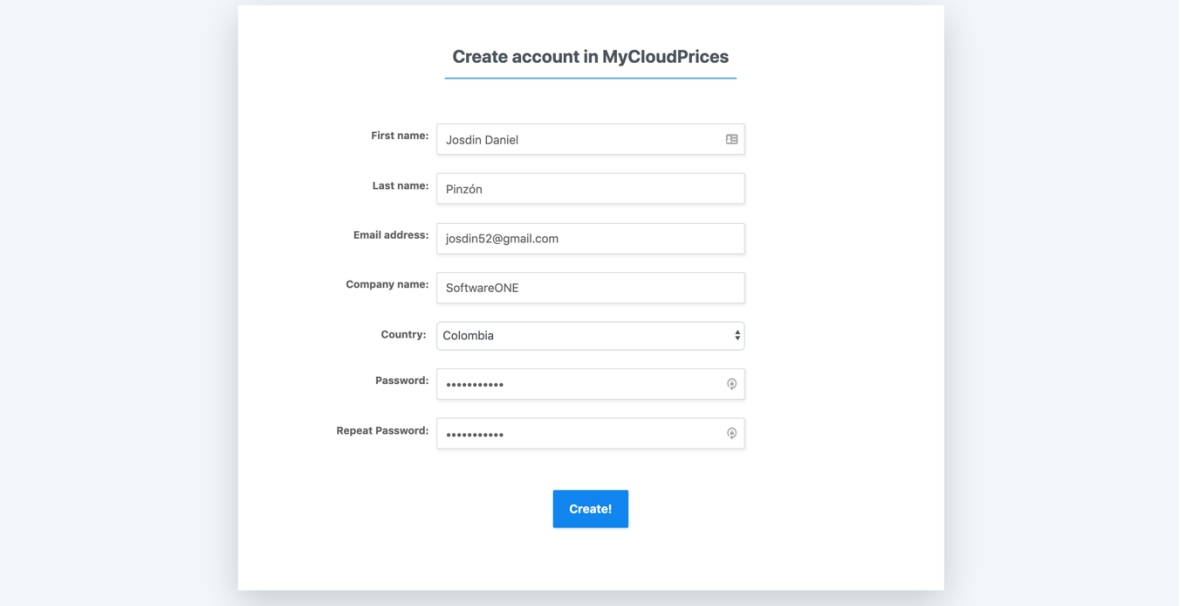


Figura 66. Prueba: Creación cuenta

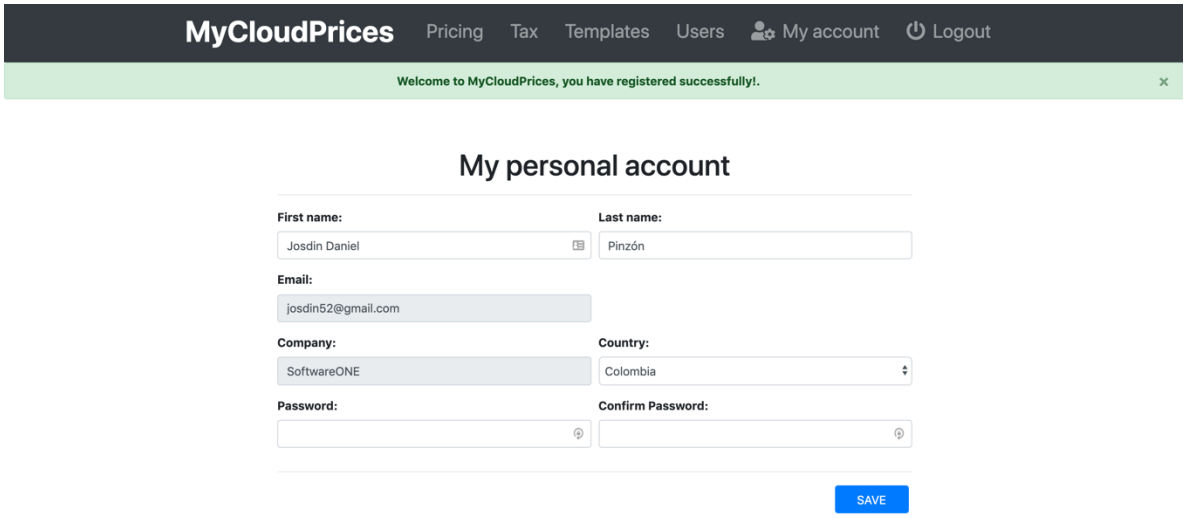


Figura 67. Prueba: Cuenta creada

- Autenticación del sistema:

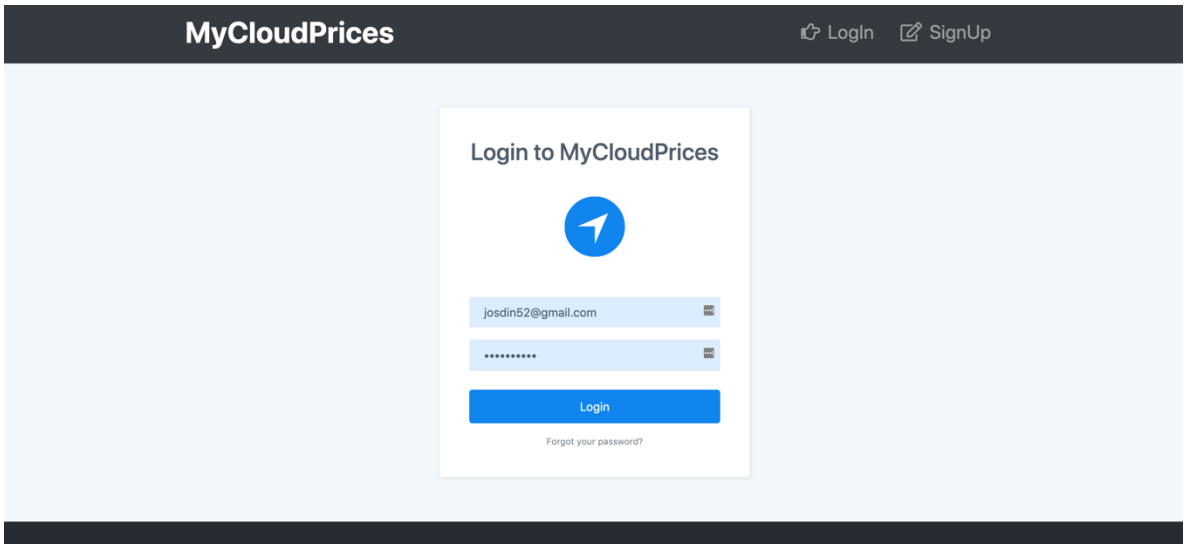


Figura 68. Prueba: Autenticación del sistema

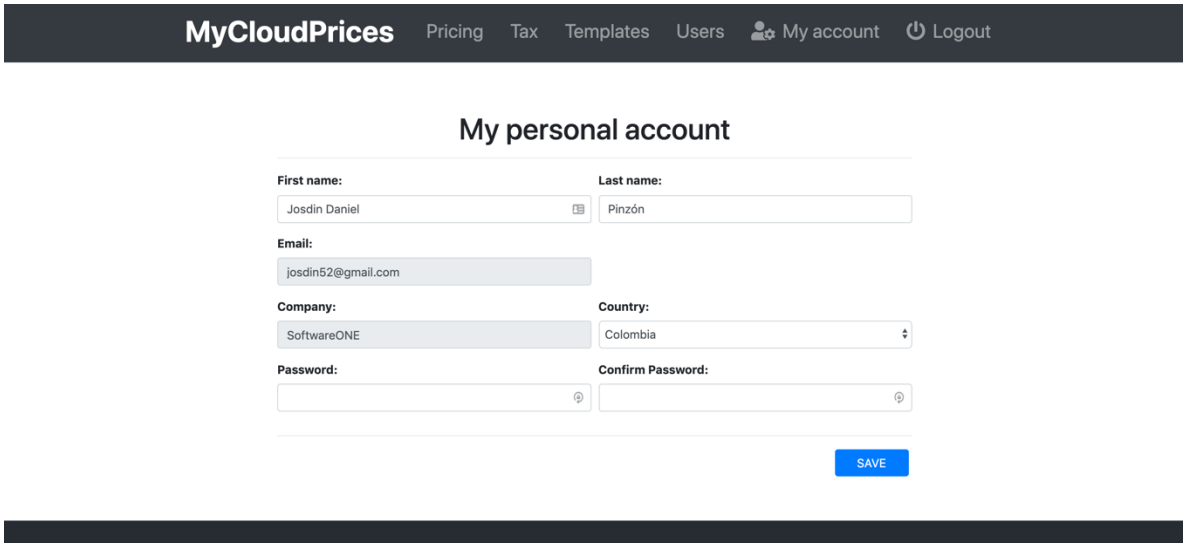


Figura 69. Prueba: Autenticación exitosa

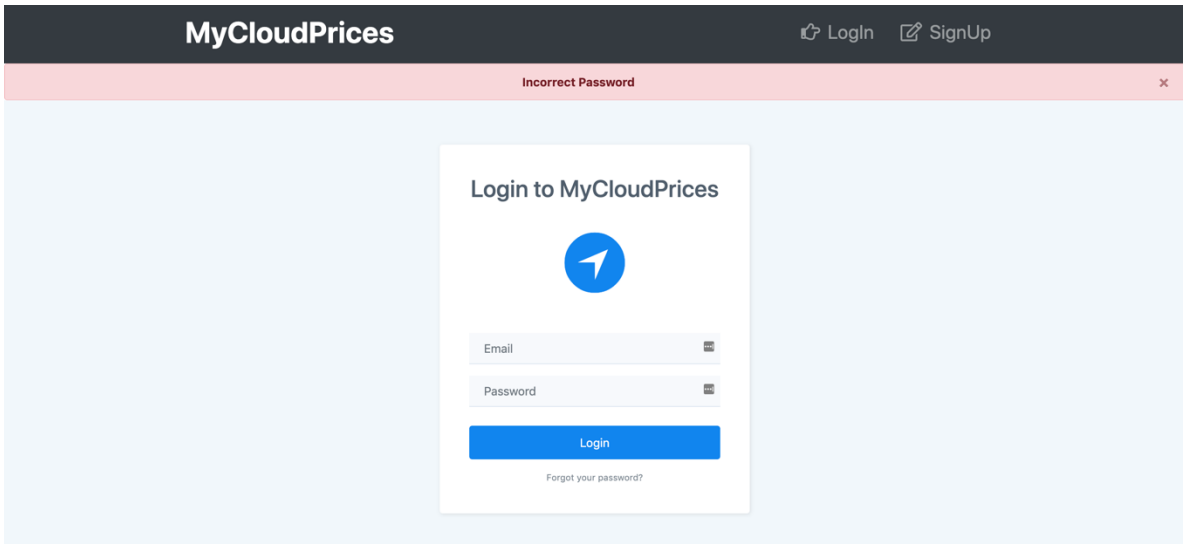


Figura 70. Prueba: Autenticación fallida

- Consulta y modificación de datos de la cuenta.

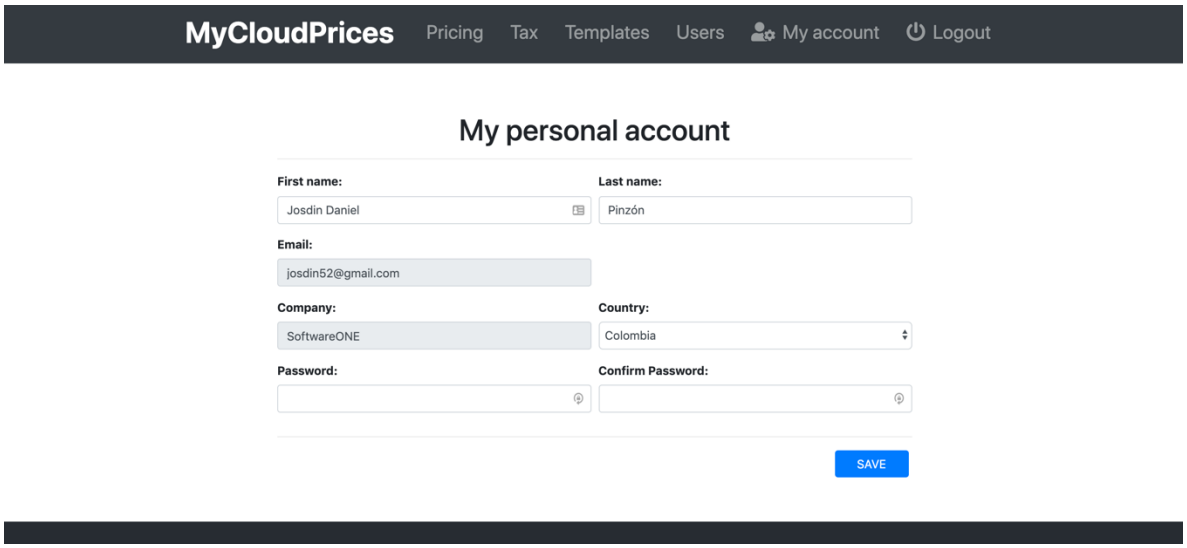


Figura 71. Prueba: Consulta datos cuenta

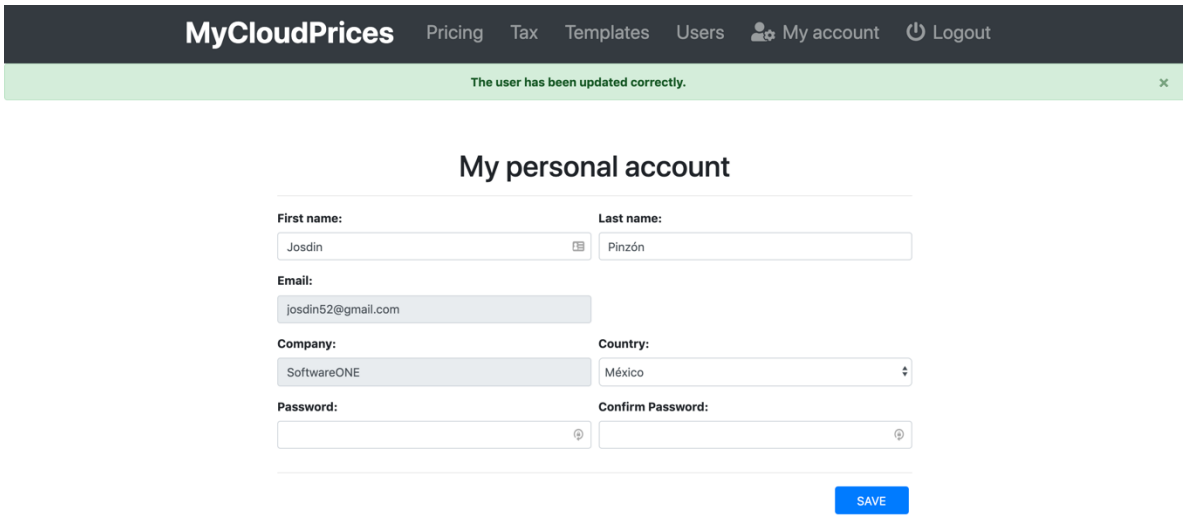


Figura 72. Prueba: Modificación datos cuenta

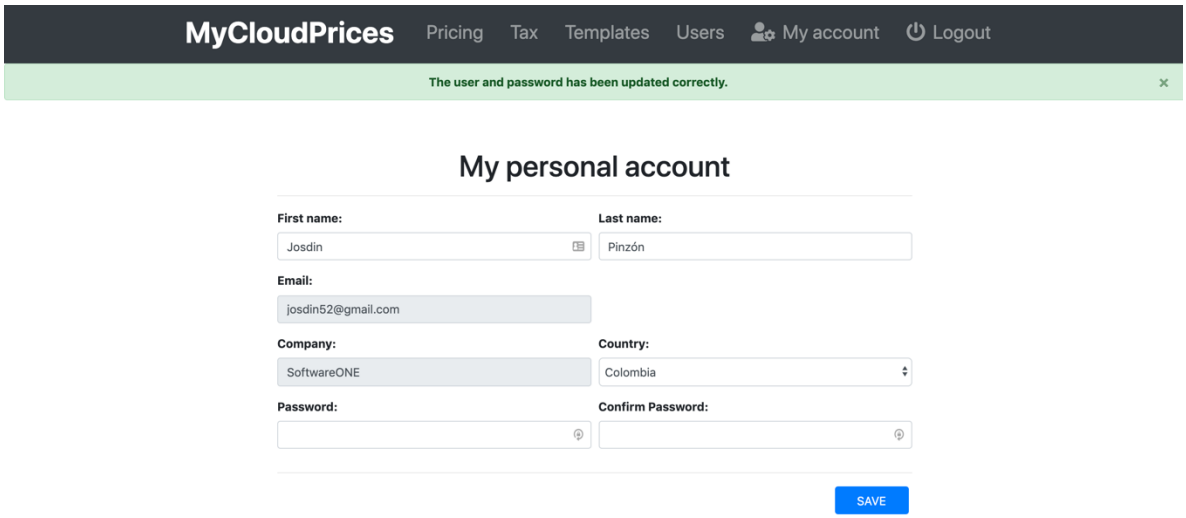


Figura 73. Prueba: Modificación contraseña cuenta

- Crear un usuario

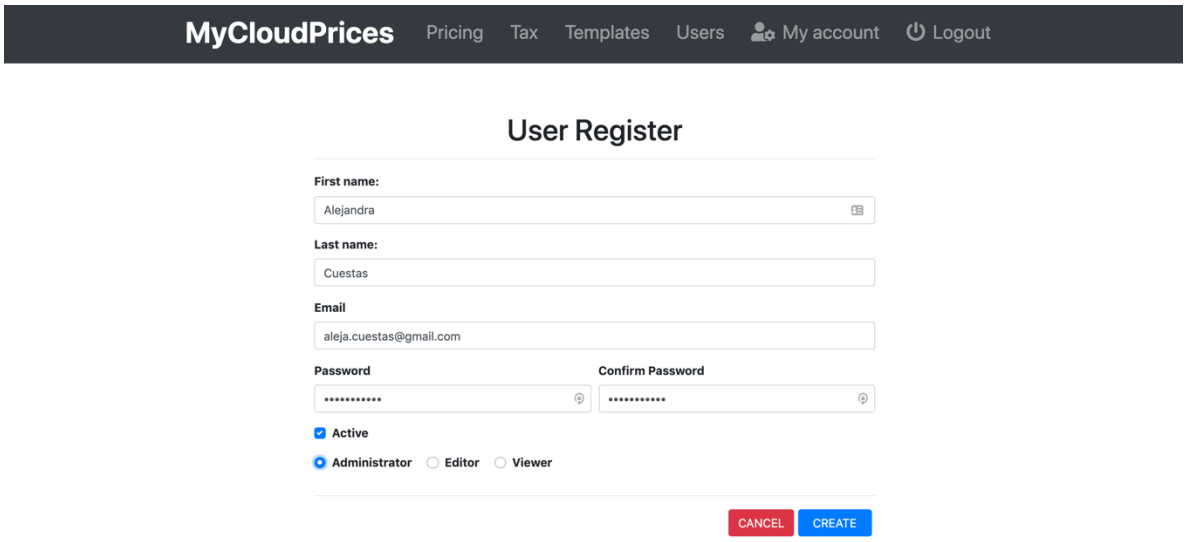


Figura 74. Prueba: Crear usuario

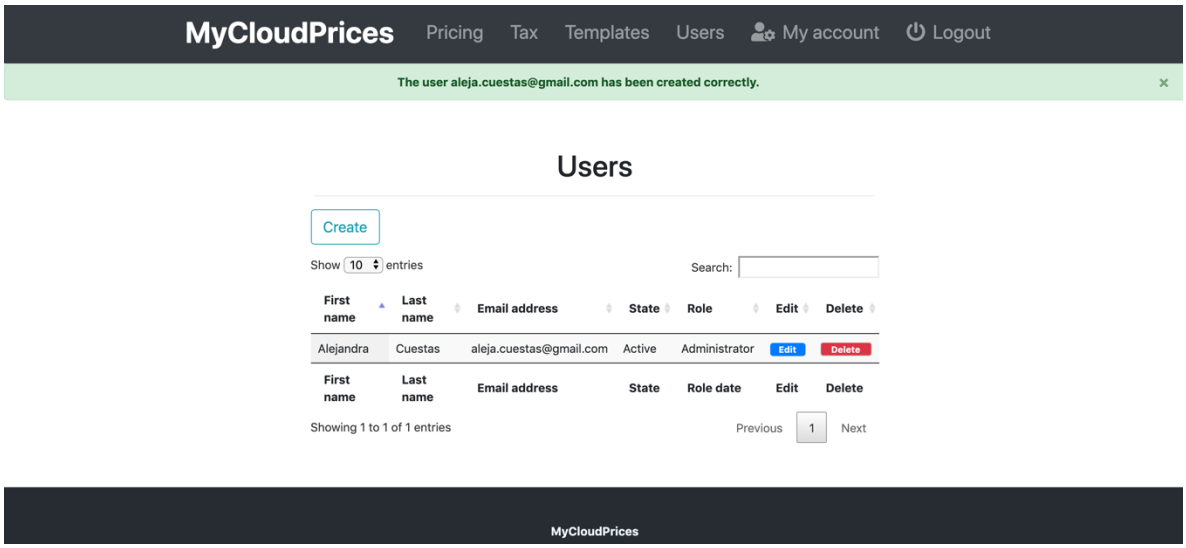


Figura 75. Prueba: Mensaje usuario creado

- Consultar listado de usuarios

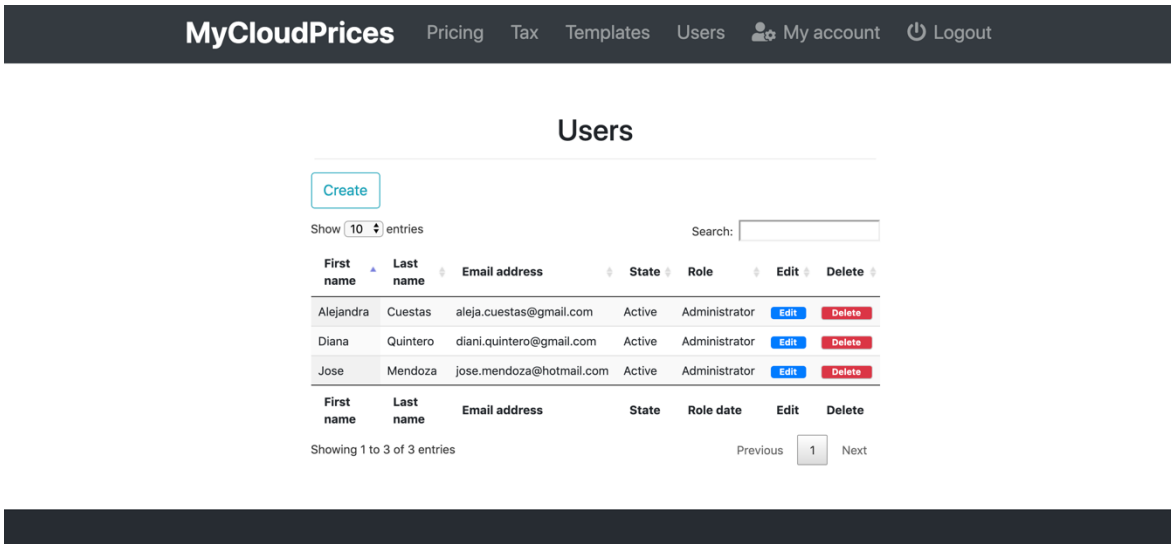


Figura 76. Prueba: Consultar usuarios

- Editar datos de un usuario

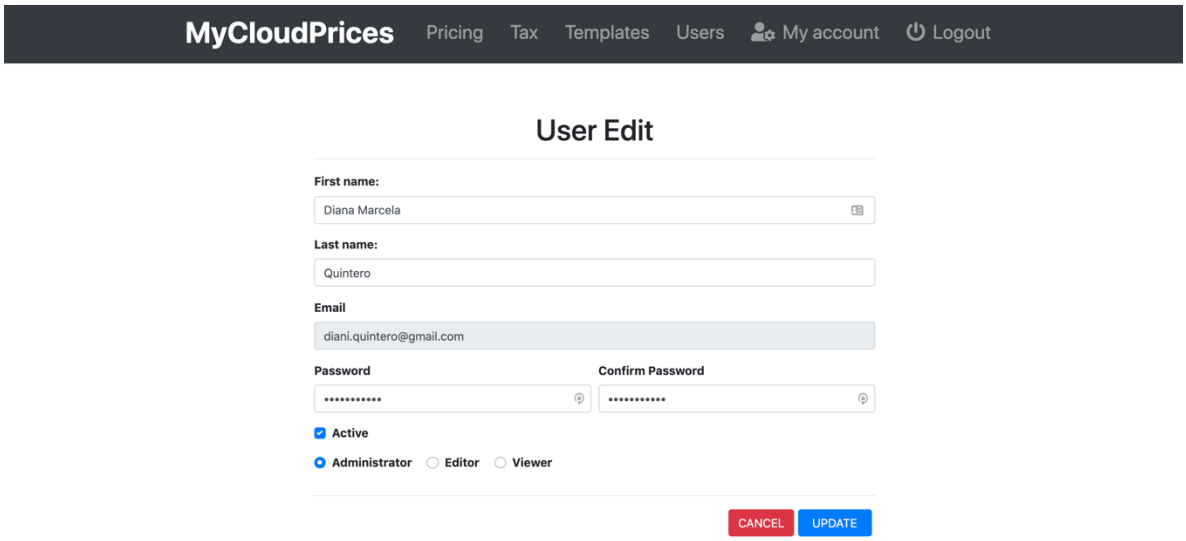


Figura 77. Prueba: Editar usuarios

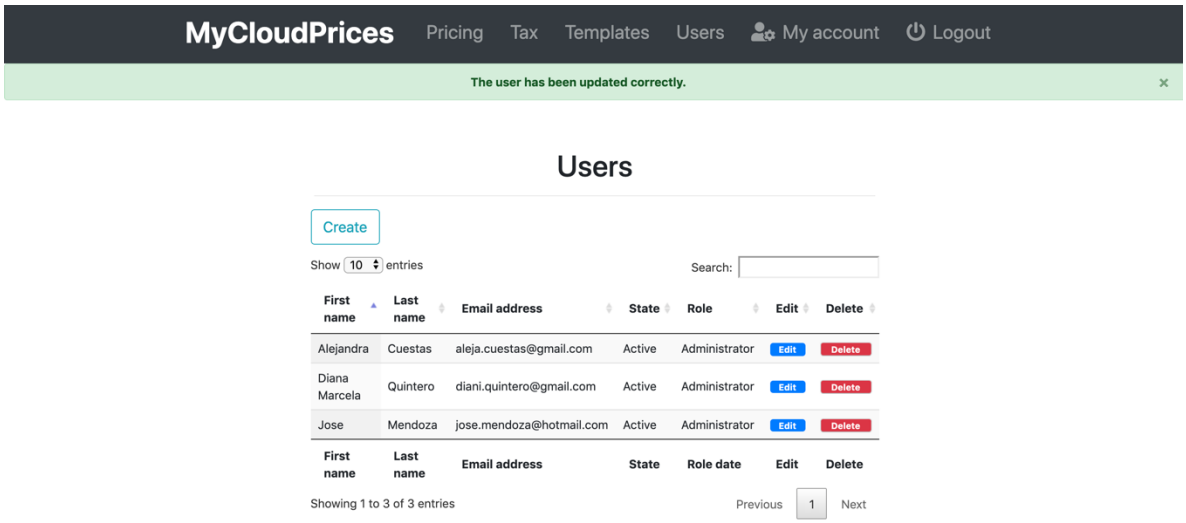


Figura 78. Prueba: Mensaje edición usuario exitosa

- Eliminar un usuario

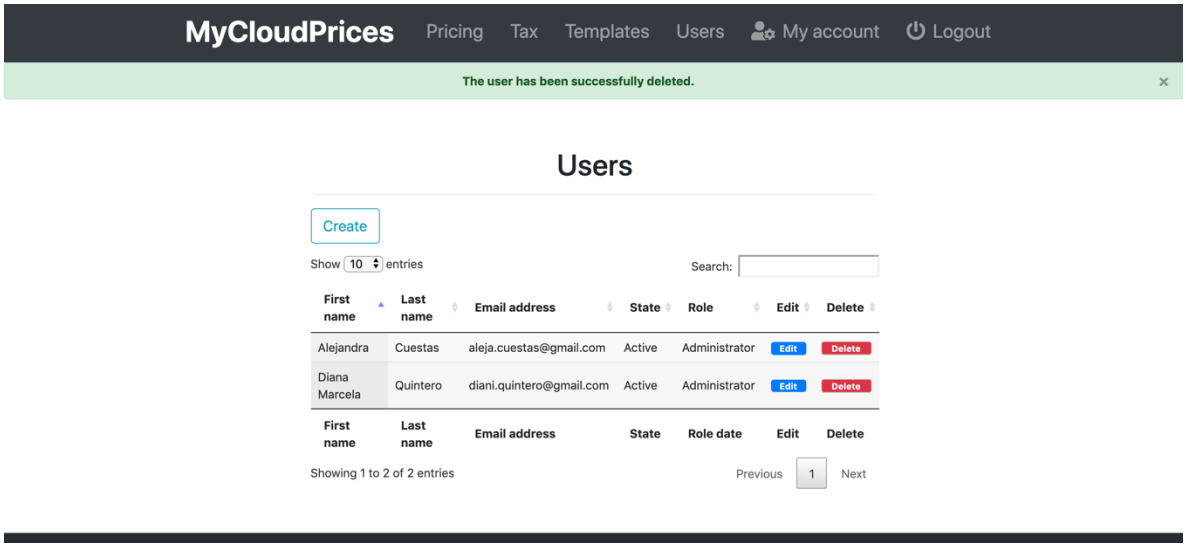


Figura 79. Prueba: Mensaje eliminación usuario exitosa

- Crear un impuesto

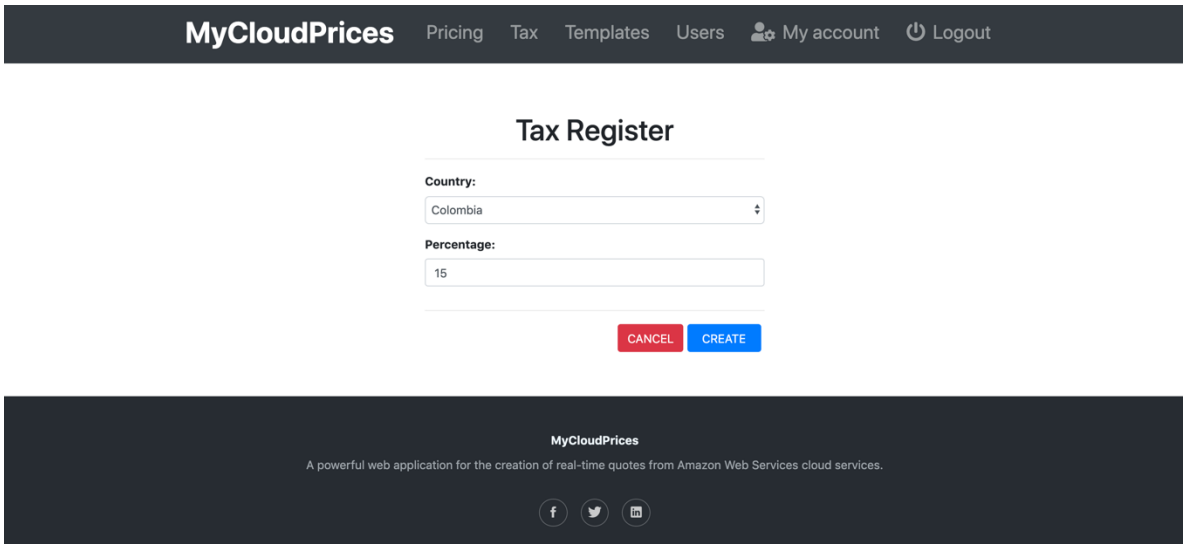


Figura 80. Prueba: Creación impuesto

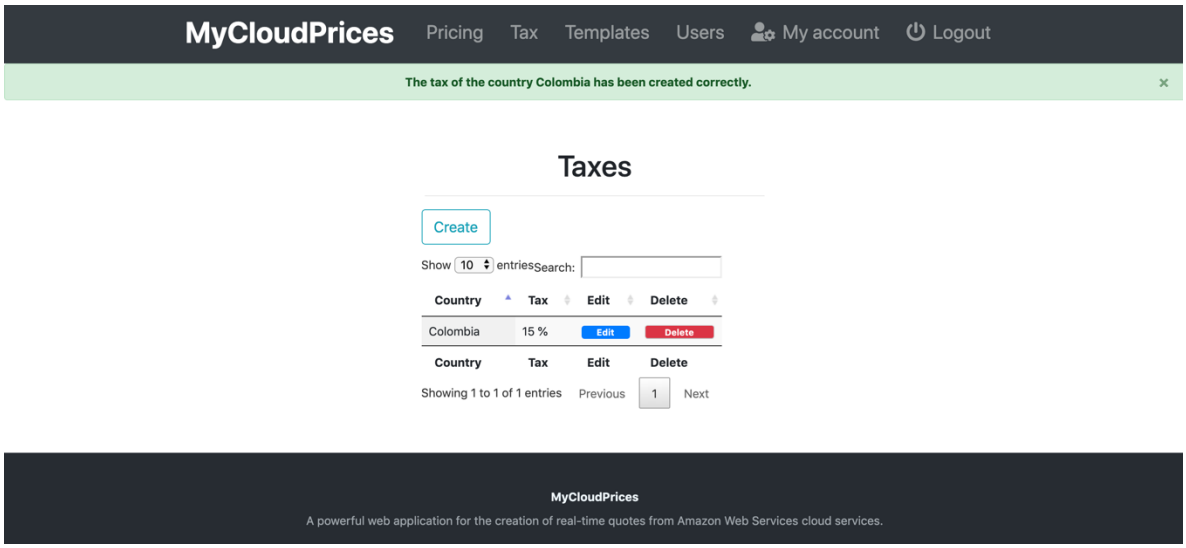


Figura 81. Prueba: Mensaje creación impuesto exitosa

- Consultar listado de impuestos

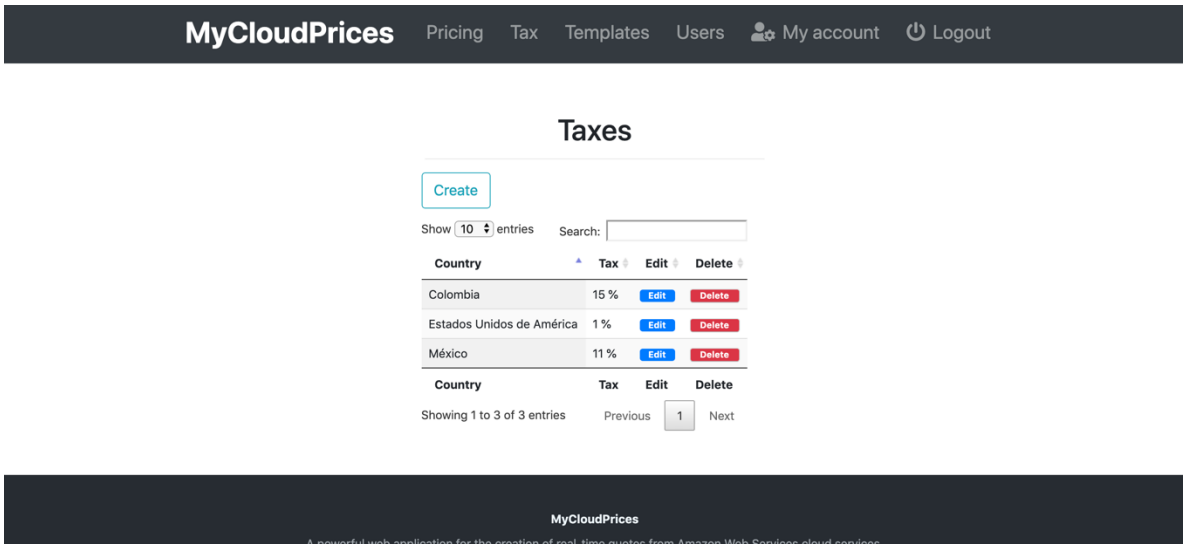


Figura 82. Prueba: Listado impuestos

- Editar datos de un impuesto

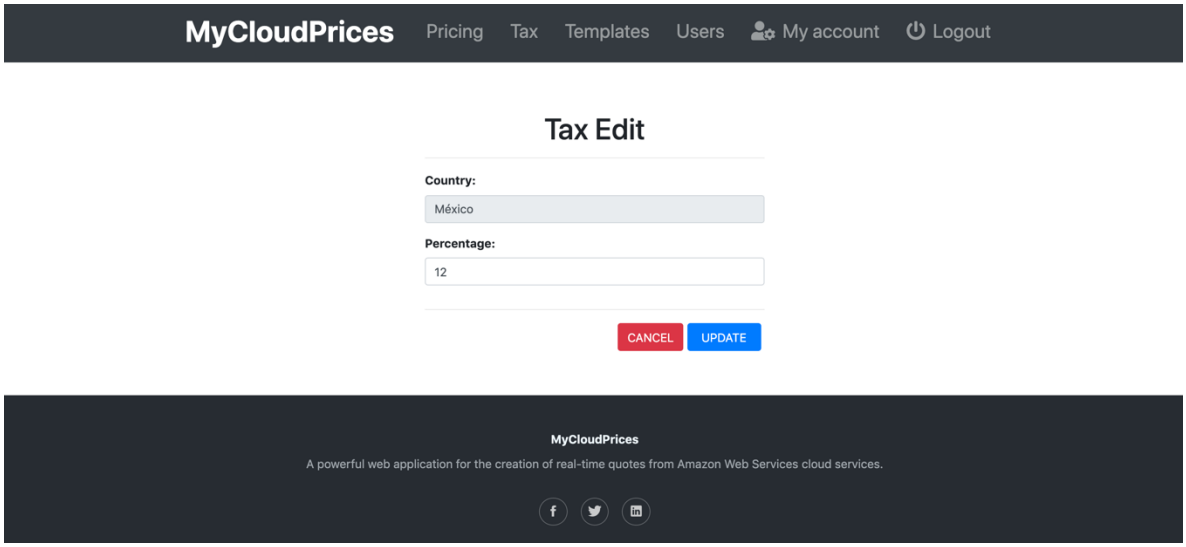


Figura 83. Prueba: Editar impuesto

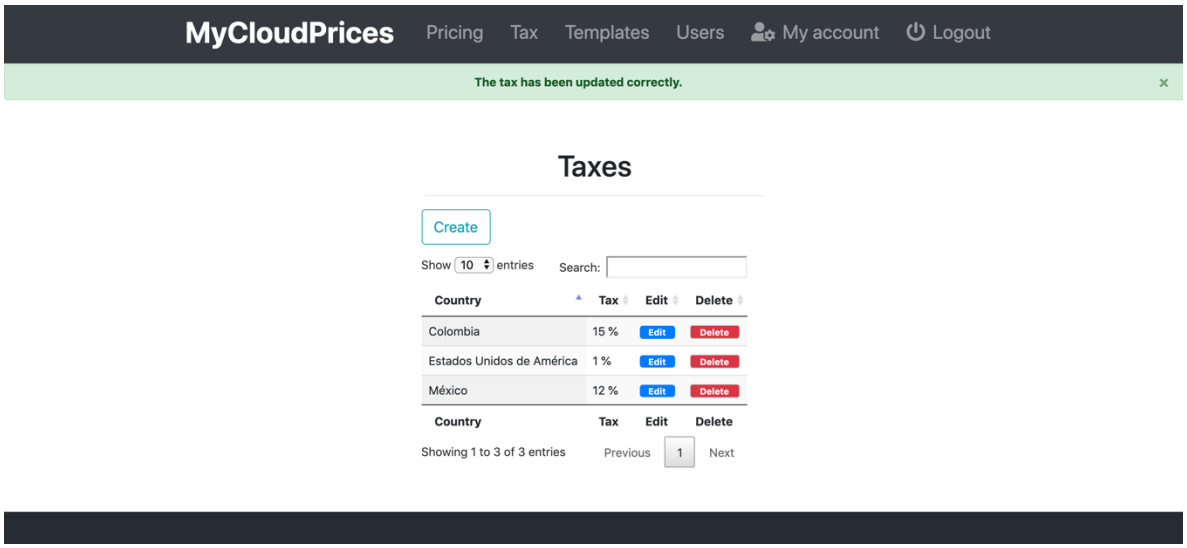


Figura 84. Prueba: Mensaje edición impuesto exitosamente

- Eliminar un impuesto

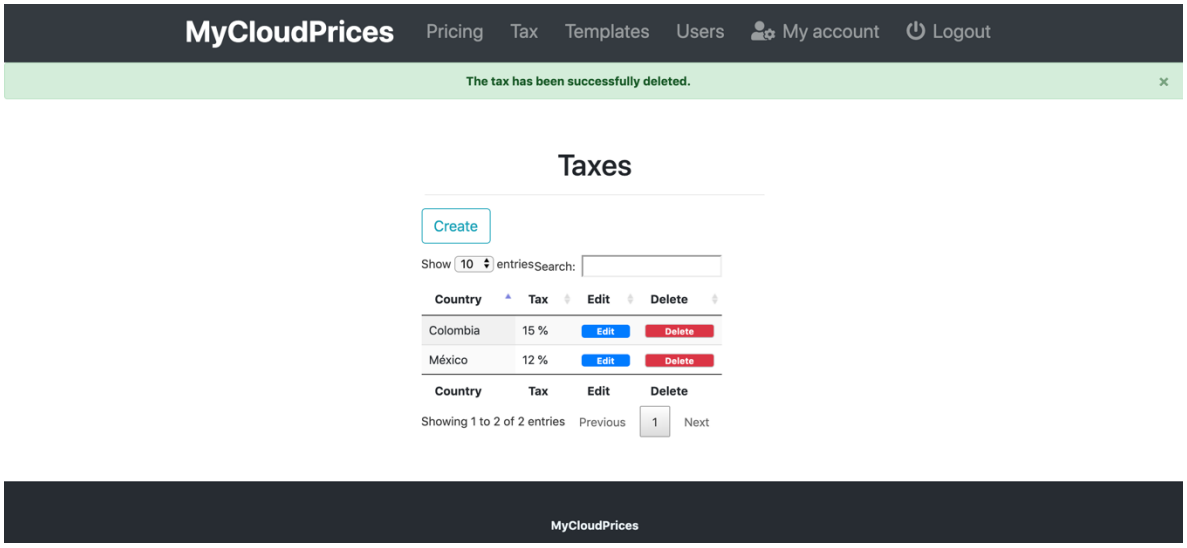


Figura 85. Prueba: Mensaje eliminación impuesto exitosamente

12. TRABAJOS FUTUROS

El prototipo se ha presentado a una empresa de consultoría generando interés por parte de los socios de la compañía, teniendo como plan ofrecer capacidad de talento humano y capacitación para agregar nuevas características y así, mejorar constantemente el desarrollo y sostenibilidad del software.

Algunas características adicionales consisten en agregar más servicios de infraestructura de AWS a la estimación de precios, como el servicio de Base de datos (Amazon RDS), el servicio de contenedores administrados (Amazon ECS y Amazon EKS) y el servicio de Backup (AWS Backup).

Finalmente, lo que se pretende con este prototipo web es convertirlo en una aplicación empresarial muy completa y funcional para aquellas empresas que brindan consultoría de infraestructura nube en Amazon Web Services.

13. CONCLUSIONES

Se concluye que el prototipo web logra tener un alcance y cubre las necesidades básicas que surgen en las empresas partners de Amazon Web Services, puesto que con la llegada de las nuevas tecnologías es necesario implementar procesos dinámicos y de fácil acceso que le permitan a las empresas atender a los clientes y brindar la asesoría especializada en cuanto a costos y servicios que presta la nube. Es por esto que para la implementación de proyectos se sugiere utilizar servicios cloud que ese encuentran al alcance inmediato y ayudan a las empresas a ahorrar tiempo y dinero en la implementación y despliegue de la infraestructura de manera rápida y eficiente, motivo por el cual el prototipo web se convierte en una herramienta fundamental que apoya en la etapa de creación de propuestas y cotizaciones en línea, dando oportunidad en el cierre efectivo de las ventas y ahorrando tiempo vital a los colaboradores que pueden invertir en tareas que tengan más prioridad e importancia para el negocio.

BIBLIOGRAFÍA

[1]. Amazon Web Services, Inc., “Servicios de computación en la nube Colombia | AWS”, 2019. [En línea]. Disponible en: https://aws.amazon.com/es/colombia/. [Último acceso: abril 2019].

[2]. Amazon Web Services, Inc., “Precios de los servicios de la nube | AWS”, 2019. [En línea]. Disponible en: https://aws.amazon.com/es/pricing/. [Último acceso: abril 2019].

[3]. Amazon Web Services, Inc., “Simple Monthly Calculator”, 2019. [En línea]. Disponible en: https://calculator.s3.amazonaws.com/index.html. [Último acceso: abril 2019].

[4]. Amazon Web Services, Inc., “What is Cloud Computing? - Amazon Web Services”, 2019. [En línea]. Disponible en: https://aws.amazon.com/es/what-is-aws/. [Último acceso: abril 2019].

[5]. Scrumstudy.com, “Una guía para el CUERPO DE CONOCIMIENTO DE SCRUM | GUÍA SBOK – Tercera Edición”, 2019. [En línea]. Disponible en: https://www.scrumstudy.com/SBOK/SCRUMstudy-SBOK-Guide-3rd-edition-spanish.pdf. [Último acceso: abril 2019].

[6]. V. C. J. Demetrio, “Desarrollo en Cascada (Waterfall) VS Desarrollo Agile-SCRUM,” 2013. [En línea]. Disponible en: http://www.northware.mx/wp-content/uploads/2013/04/Desarrollo-cascada-vs-Desarrollo-Agile.pdf. [Último acceso: abril 2019].

[7]. SCRUM TECHNOTIVIT. “Curso de Scrum” 2018. [En línea]. Disponible en la intranet de http://intranet-latam.tivit.com/. [Último acceso: febrero 2018].

[8]. Amazon Web Services, Inc., “Introduction to DevOps on AWS”, 2014 [En línea]. Disponible en: https://d1.awsstatic.com/whitepapers/AWS_DevOps.pdf. [Último acceso: Abril 2019].

[9]. DEVOPS TECHNOTIVIT. “Curso de Devops” 2018. [En línea]. Disponible en la intranet de http://intranet-latam.tivit.com/. [Último acceso: Febrero 2018].

[10]. Carlos Azaustre, “Desarrollo Full Stack JavaScript”, 2014 [En línea]. Disponible en: https://carlosazaustre.es/desarrollo-full-stack-javascript-tambien-conocido-como-mean/. [Último acceso: Abril 2019].

[11]. Blog.faztweb.com, “Curso de Express.js, el Framework de Nodejs | ¿Qué es Express?”, 2017 [En línea]. Disponible en: http://blog.faztweb.com/2017/09/expressjs-el-framework-de-nodejs-que-es.html. [Último acceso: Abril 2019].

[12]. OpenWebinars.net, “Por qué usar Terraform?”, 2017. [En línea]. Disponible en: https://openwebinars.net/blog/por-que-usar-terraform/. [Último acceso: Abril 2019].

[13]. Grupoeducare.com, "Infraestructura Tecnológica", 2019. [En línea]. Disponible en: https://www.grupoeducare.com/dcempresarial/index.php/2-principal/38-infraestructura-tecnologica. [Último acceso: Abril 2019].

[14]. Que es arquitectura tecnológica

[15]. Amazon Web Services, Inc., “Tipos de Cloud computing”, 2019. [En línea]. Disponible en: https://aws.amazon.com/es/types-of-cloud-computing/. [Último acceso: Abril 2019].

[16]. Amazon Web Services, Inc., “Amazon EC2” 2018. [En línea]. Disponible en: https://aws.amazon.com/es/ec2/. [Último acceso: 1 Mayo 2018].

[17]. Amazon Web Services, Inc., “Amazon S3” 2018. [En línea]. Disponible en: https://aws.amazon.com/es/s3/. [Último acceso: 1 Mayo 2018].

[18]. Amazon Web Services, Inc., “Amazon Elastic Block Store,” 2018. [En línea]. Disponible en: https://aws.amazon.com/es/ebs/. [Último acceso: 1 Mayo 2018].

[19]. Amazon Web Services, Inc., “Amazon Relational Database Services (RDS),” 2018. [En línea]. Disponible en: https://aws.amazon.com/es/rds/. [Último acceso: 1 Mayo 2018].

[20]. Amazon Web Services, Inc., “AWS Elastic Load Balancing”, 2018 [En línea]. Disponible en: https://aws.amazon.com/elasticloadbalancing/. [Último acceso: 1 mayo 2018].

[21]. Amazon Web Services, Inc., “AWS Auto Scaling”, 2018 [En línea]. Disponible en: https://aws.amazon.com/es/autoscaling/. [Último acceso: 1 mayo 2018].

[22]. Amazon Web Services, Inc., “AWS | Lambda”, 2018 [En línea]. Disponible en: https://aws.amazon.com/es/lambda/. [Último acceso: 1 mayo 2018].

[23]. Amazon Web Services, Inc., “Amazon DynamoDB – NoSQL Cloud Database Service”, 2018 [En línea]. Disponible en: https://aws.amazon.com/dynamodb/. [Último acceso: 1 mayo 2018].

[24]. Amazon Web Services, Inc., “Amazon CloudWatch – Cloud & Network Monitoring Services”, 2018 [En línea]. Disponible en: https://aws.amazon.com/cloudwatch/. [Último acceso: 1 mayo 2018].

[25]. Amazon Web Services, Inc., “AWS CloudTrail”, 2018 [En línea]. Disponible en: https://aws.amazon.com/es/cloudtrail/. [Último acceso: 1 mayo 2018].

[26]. AWS News Blog, “New – AWS Price List API”, 2018 [En línea]. Disponible en: https://aws.amazon.com/es/blogs/aws/new-aws-price-list-api/. [Último acceso: 1 mayo 2018].

[27]. Economipedia, “Cotización,” 2018. [En línea]. Disponible en: http://economipedia.com/definiciones/cotizacion.html. [Último acceso: 1 mayo 2018].

[28]. Si-emprendemos (Blog), “Qué son los partners y como nos ayudan a los emprendedores,” 2014. [En línea]. Disponible en: https://si-emprendemos.blogspot.com.co/2014/01/que-son-los-partners-y-como-nos-ayudan.html. [Último acceso: 1 mayo 2018].

[29]. Hewlett packard enterprise, “¿Qué es la infraestructura como código?,” 2018. [En línea]. Disponible en: https://www.hpe.com/lamerica/es/what-is/infrastructure-as-code.html. [Último acceso: 1 mayo 2018].

[30]. HASHICORP, “Cloud Infrastructure Automation,” 2018. [En línea]. Disponible en: https://www.terraform.io/. [Último acceso: 1 mayo 2018].

[31]. Ramos,J. Programacionymas.com, “¿Qué es el Product Backlog?”, 2019. [En línea]. Disponible en: https://programacionymas.com/blog/scrum-product-backlog. [Último acceso: abril 2019].

[32]. Que es un sprint

[33]. Diegoacosta.net, “Scrum con Jira – Relación entre Épicas, Historias y Tareas Técnicas. | | Blog Personal Diego Acosta", 2019. [en línea]. Disponible en:

http://diegoacosta.net/blog/2017/12/10/scrum-con-jira-relacion-entre-epicas-historias-y-tareas-tecnicas/. [Último acceso: Abril 2019].

ANEXOS

Anexo A

Infrastructure.tf

```
# Initial configuration AWS
provider "aws" {
  access_key  = "${var.access_key}"
  secret_key  = "${var.secret_key}"
  region      = "${var.region}"
}

# * * * * * * * * * Networking * * * * * * * * *
# Create VPC
resource "aws_vpc" "vpc-mcp" {
  cidr_block           = "${var.cidrs_vpc[0]}"
  enable_dns_hostnames  = true
  tags {
    Name  = "vpc-mcp"
    Env   = "prod"
  }
}

# Create subnet public 1
resource "aws_subnet" "public_subnet-mcp" {
  vpc_id                  = "${aws_vpc.vpc-mcp.id}"
  cidr_block              = "${var.cidrs_vpc[1]}"
  availability_zone       = "${var.az[0]}"
  map_public_ip_on_launch = true
  tags {
    Name  = "public_subnet-mcp"
    Env   = "prod"
  }
}

# Create subnet public 2
resource "aws_subnet" "public_subnet2-mcp" {
  vpc_id                  = "${aws_vpc.vpc-mcp.id}"
  cidr_block              = "${var.cidrs_vpc[2]}"
  availability_zone       = "${var.az[1]}"
  map_public_ip_on_launch = true
  tags {
    Name  = "public_subnet2-mcp"
    Env   = "prod"
  }
```

```
}

# Create subnet private 1
resource "aws_subnet" "private_subnet-mcp" {
  vpc_id              = "${aws_vpc.vpc-mcp.id}"
  cidr_block          = "${var.cidrs_vpc[3]}"
  availability_zone   = "${var.az[0]}"
  tags {
    Name  = "private_subnet-mcp"
    Env   = "prod"
  }
}

# Create subnet private 2
resource "aws_subnet" "private_subnet2-mcp" {
  vpc_id              = "${aws_vpc.vpc-mcp.id}"
  cidr_block          = "${var.cidrs_vpc[4]}"
  availability_zone   = "${var.az[1]}"
  tags {
    Name  = "private_subnet2-mcp"
    Env   = "prod"
  }
}

# Create internet gateway
resource "aws_internet_gateway" "igw-mcp"{
  vpc_id  = "${aws_vpc.vpc-mcp.id}"
  tags {
    Name  = "igw-mcp"
    Env   = "prod"
  }
}

# Create public route table
resource "aws_route_table" "public-rt-mcp" {
  vpc_id        = "${aws_vpc.vpc-mcp.id}"
  route {
    cidr_block  = "0.0.0.0/0"
    gateway_id  = "${aws_internet_gateway.igw-mcp.id}"
  }
  tags {
    Name  = "public-rt-mcp"
    Env   = "prod"
  }
}
```

```
# Association public subnet 1 and public route table
resource "aws_route_table_association" "public-rta" {
  subnet_id      = "${aws_subnet.public_subnet-mcp.id}"
  route_table_id = "${aws_route_table.public-rt-mcp.id}"
}

# Association public subnet 2 and public route table
resource "aws_route_table_association" "public-rta2" {
  subnet_id      = "${aws_subnet.public_subnet2-mcp.id}"
  route_table_id = "${aws_route_table.public-rt-mcp.id}"
}

# Create private route table 1
resource "aws_route_table" "private-rt-mcp" {
  vpc_id            = "${aws_vpc.vpc-mcp.id}"
  route {
    cidr_block      = "0.0.0.0/0"
    nat_gateway_id  = "${aws_nat_gateway.nat-gw-mcp.id}"
  }
  tags {
    Name  = "private-rt-mcp"
    Env   = "prod"
  }
}

# Association private subnet 1 and private route table 1
resource "aws_route_table_association" "private-rta" {
  subnet_id      = "${aws_subnet.private_subnet-mcp.id}"
  route_table_id = "${aws_route_table.private-rt-mcp.id}"
}

# Create private route table 2
resource "aws_route_table" "private-rt2-mcp" {
  vpc_id            = "${aws_vpc.vpc-mcp.id}"
  route {
    cidr_block      = "0.0.0.0/0"
    nat_gateway_id  = "${aws_nat_gateway.nat2-gw-mcp.id}"
  }
  tags {
    Name  = "private-rt2-mcp"
    Env   = "prod"
  }
}
```

```
# Association private subnet 2 and private route table 1
resource "aws_route_table_association" "private-rta2" {
  subnet_id      = "${aws_subnet.private_subnet2-mcp.id}"
  route_table_id = "${aws_route_table.private-rt2-mcp.id}"
}

# Create IP for NAT Gateway 1
resource "aws_eip" "eip-natgateway-mcp" {
  vpc = true
}

# Create NAT Gateway 1
resource "aws_nat_gateway" "nat-gw-mcp" {
  allocation_id = "${aws_eip.eip-natgateway-mcp.id}"
  subnet_id     = "${aws_subnet.public_subnet-mcp.id}"
  tags = {
    Name  = "nat-gw-mcp"
    Env   = "prod"
  }
}

# Create IP for NAT Gateway 2
resource "aws_eip" "eip-natgateway2-mcp" {
  vpc = true
}

# Create NAT Gateway 2
resource "aws_nat_gateway" "nat2-gw-mcp" {
  allocation_id = "${aws_eip.eip-natgateway2-mcp.id}"
  subnet_id     = "${aws_subnet.public_subnet2-mcp.id}"
  tags = {
    Name  = "nat2-gw-mcp"
    Env   = "prod"
  }
}

# Create security groups load-balancer
resource "aws_security_group" "lb-sg-mcp" {
  name        = "lb-sg-mcp"
  description = "Load Balancer traffic"
  vpc_id      = "${aws_vpc.vpc-mcp.id}"
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
```

```
    cidr_blocks = ["0.0.0.0/0"]
  }
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
  tags {
    Name  = "lb-sg-mcp"
    Env   = "prod"
  }
}

# Create security groups bastion-host
resource "aws_security_group" "bh-sg-mcp" {
  name        = "bh-sg-mcp"
  description = "Secure connection SSH for instances"
  vpc_id      = "${aws_vpc.vpc-mcp.id}"
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["186.155.132.115/32"] // Change for my ip address
  }
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
  tags {
    Name  = "bh-sg-mcp"
    Env   = "prod"
  }
}

# Create security groups Web
resource "aws_security_group" "web-sg-mcp" {
  name        = "web-sg-mcp"
  description = "Allow inbound web traffic"
  vpc_id      = "${aws_vpc.vpc-mcp.id}"
  ingress {
    from_port       = 22
    to_port         = 22
```

```
    protocol        = "tcp"
    security_groups = ["${aws_security_group.bh-sg-mcp.id}"]
  }
  ingress {
    from_port       = 80
    to_port         = 80
    protocol        = "tcp"
    security_groups = ["${aws_security_group.lb-sg-mcp.id}"]
  }
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
  tags {
    Name  = "web-sg-mcp"
    Env   = "prod"
  }
}

# Create security groups database
resource "aws_security_group" "db-sg-mcp" {
  name        = "db-sg-mcp"
  description = "Secure connection database for instances"
  vpc_id      = "${aws_vpc.vpc-mcp.id}"

  ingress {
    from_port       = 27017
    to_port         = 27017
    protocol        = "tcp"
    security_groups = ["${aws_security_group.web-sg-mcp.id}"]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags {
    Name  = "db-sg-mcp"
    Env   = "prod"
  }
```

```
}

# * * * * * * * * * IAM permissions * * * * * * * * *

resource "aws_iam_role" "app-role" {
  name               = "app-role"
  assume_role_policy  = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "sts:AssumeRole",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Effect": "Allow",
      "Sid": ""
    }
  ]
}
EOF
tags {
    Name  = "app-role"
    Env   = "prod"
  }
}

resource "aws_iam_instance_profile" "app-profile" {
  name = "app-profile"
  role = "${aws_iam_role.app-role.name}"
}

resource "aws_iam_role_policy" "app-role_policy" {
    name    = "app-role_policy"
    role    = "${aws_iam_role.app-role.id}"
    policy  = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:BatchGet*",
        "codecommit:Get*",
        "codecommit:Describe*",
```

```
        "codecommit:List*",
        "codecommit:GitPull"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CloudWatchEventsCodeCommitRulesReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",
        "events:ListTargetsByRule"
      ],
      "Resource": "arn:aws:events:*:*:rule/codecommit*"
    },
    {
      "Sid": "SNSSubscriptionAccess",
      "Effect": "Allow",
      "Action": [
        "sns:ListTopics",
        "sns:ListSubscriptionsByTopic",
        "sns:GetTopicAttributes"
      ],
      "Resource": "*"
    },
    {
      "Sid": "LambdaReadOnlyListAccess",
      "Effect": "Allow",
      "Action": [
        "lambda:ListFunctions"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IAMReadOnlyListAccess",
      "Effect": "Allow",
      "Action": [
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}
EOF
}
```

```
# * * * * * * * * * Database NoSQL * * * * * * * * *

resource "aws_docdb_cluster" "docdb-mcp" {
  cluster_identifier      = "my-docdb-cluster"
  engine                  = "docdb"
  master_username         = "${var.userdb}"
  master_password         = "${var.passdb}"
  backup_retention_period = 5
  preferred_backup_window = "05:00-06:00"
  skip_final_snapshot     = true
  apply_immediately = true
  availability_zones = ["${var.az[0]}", "${var.az[1]}"]
  db_subnet_group_name = "${aws_docdb_subnet_group.docdb-sg-mcp.id}"
                                    db_cluster_parameter_group_name                    =
"${aws_docdb_cluster_parameter_group.docdb-pg-mcp.id}"
  vpc_security_group_ids = ["${aws_security_group.db-sg-mcp.id}"]
  tags = {
    Name = "docdb-mcp"
    Env = "prod"
  }
}

resource "aws_docdb_cluster_instance" "docdb-cluster-mcp" {
  count             = 1
  identifier        = "my-docdb-cluster-${count.index}"
  cluster_identifier = "${aws_docdb_cluster.docdb-mcp.id}"
  instance_class     = "db.r4.large"
  apply_immediately = true
  tags = {
    Name = "docdb-cluster-mcp"
    Env = "prod"
  }
}

resource "aws_docdb_subnet_group" "docdb-sg-mcp" {
  name       = "docdb-sg-mcp"
  description = "docdb cluster subnet group"
                      subnet_ids            =           ["${aws_subnet.public_subnet-mcp.id}",
"${aws_subnet.public_subnet2-mcp.id}"]
  tags = {
    Name = "docdb-sg-mcp"
    Env = "prod"
  }
}
```

```
resource "aws_docdb_cluster_parameter_group" "docdb-pg-mcp" {
  family      = "docdb3.6"
  name        = "docdb-pg-mcp"
  description = "docdb cluster parameter group"
  parameter {
    name  = "tls"
    value = "enabled"
  }
  tags = {
    Name = "docdb-pg-mcp"
    Env = "prod"
  }
}

# * * * * * * * * * Compute * * * * * * * * *

# Create bastion host instance 1
resource "aws_instance" "bastionhost-mcp" {
  ami                    = "${var.ec2_info[0]}"
  instance_type          = "${var.ec2_info[1]}"
  availability_zone      = "${var.az[0]}"
  key_name               = "${var.key_name}"
  vpc_security_group_ids  = ["${aws_security_group.bh-sg-mcp.id}"]
  subnet_id              = "${aws_subnet.public_subnet-mcp.id}"
  tags {
    Name  = "bastionhost-mcp"
    Env   = "prod"
  }
}

# Assign elastic ip to bastion host instance 1
resource "aws_eip" "ip-bastionhost-mcp" {
  instance    = "${aws_instance.bastionhost-mcp.id}"
  depends_on  = ["aws_instance.bastionhost-mcp"]
  tags {
    Name  = "ip-bastionhost-mcp"
    Env   = "prod"
  }
}

# Create bastion host instance 2
resource "aws_instance" "bastionhost2-mcp" {
  ami                    = "${var.ec2_info[0]}"
  instance_type          = "${var.ec2_info[1]}"
  availability_zone      = "${var.az[1]}"
```

```
  key_name               = "${var.key_name}"
  vpc_security_group_ids  = ["${aws_security_group.bh-sg-mcp.id}"]
  subnet_id              = "${aws_subnet.public_subnet2-mcp.id}"
  tags {
    Name  = "bastionhost2-mcp"
    Env   = "prod"
  }
}

# Assign elastic ip to bastion host instance 2
resource "aws_eip" "ip-bastionhost2-mcp" {
  instance    = "${aws_instance.bastionhost2-mcp.id}"
  depends_on  = ["aws_instance.bastionhost2-mcp"]
  tags {
    Name  = "ip-bastionhost2-mcp"
    Env   = "prod"
  }
}

# Create Application Load Balancer
resource "aws_lb" "alb-mcp" {
  name               = "alb-mcp"
  internal           = false
  load_balancer_type = "application"
  security_groups    = ["${aws_security_group.lb-sg-mcp.id}"]
      subnets                                    =    ["${aws_subnet.public_subnet-mcp.id}",
"${aws_subnet.public_subnet2-mcp.id}"]
  tags {
    Name  = "alb-mcp"
    Env   = "prod"
  }
}

# Create Application Load Balancer - target group
resource "aws_lb_target_group" "alb_target_mcp" {
  name                 = "albtargetmcp"
  port                 = 80
  protocol             = "HTTP"
  deregistration_delay  = 60
  vpc_id               = "${aws_vpc.vpc-mcp.id}"
  health_check {
    healthy_threshold   = 5
    unhealthy_threshold = 2
    timeout             = 5
    interval            = 30
```

```
    path                = "/"
    protocol            = "HTTP"
    matcher             = "200-399"
  }
  stickiness {
    type = "lb_cookie"
    enabled = true
  }
  tags {
    Name  = "alb_target_mcp"
    Env   = "prod"
  }
}

# Create Application Load Balancer - listener
resource "aws_lb_listener" "alb_listener_mcp" {
  load_balancer_arn = "${aws_lb.alb-mcp.arn}"
  port              = "80"
  protocol          = "HTTP"

  default_action {
    type             = "forward"
    target_group_arn = "${aws_lb_target_group.alb_target_mcp.arn}"
  }
}

# Create Launch Configuration
resource "aws_launch_configuration" "lc-mcp" {
  name                 = "lc-mcp"
  image_id             = "${var.ec2_info[0]}"
  instance_type        = "${var.ec2_info[1]}"
  security_groups      = ["${aws_security_group.web-sg-mcp.id}"]
  key_name             = "amazonvm"
  iam_instance_profile   = "${aws_iam_instance_profile.app-profile.name}"
  user_data = <<EOF
    #! /bin/bash
    yum install -y git
    yum install -y gcc-c++ make
    curl -sL https://rpm.nodesource.com/setup_11.x | sudo -E bash -
    yum install -y nodejs
    git config --system credential.https://git-codecommit.us-east-1.amazonaws.com.helper '!aws codecommit credential-helper $@'
    git config --system credential.https://git-codecommit.us-east-1.amazonaws.com.UseHttpPath true
```

```
                                                                    git                    clone
https://git-codecommit.us-east-1.amazonaws.com/v1/repos/mycloudprices
/home/ec2-user/mycloudprices
    printf "module.exports = {\n     mongodb: {\n         URI: 'mongodburl'\n     }\n};" >
/home/ec2-user/mycloudprices/src/keys.js
                                                                    sed                    -i
"s+mongodburl+mongodb://${var.userdb}:${var.passdb}@${aws_docdb_cluster.do
cdb-mcp.endpoint}:${var.port}/${var.databasename}?ssl=true&replicaSet=rs0+g"
/home/ec2-user/mycloudprices/src/keys.js
   sudo chown -R ec2-user /home/ec2-user/mycloudprices/
   sudo chgrp -R ec2-user /home/ec2-user/mycloudprices/
   cd /home/ec2-user/mycloudprices
   npm run start
  EOF
                        depends_on            =            ["aws_docdb_cluster.docdb-mcp",
"aws_docdb_cluster_instance.docdb-cluster-mcp"]
  lifecycle {
    create_before_destroy = true
  }
}
## Creating AutoScaling Group
resource "aws_autoscaling_group" "asg-mcp" {
  name                 = "asg-mcp"
  launch_configuration  = "${aws_launch_configuration.lc-mcp.id}"
                                vpc_zone_identifier                                    =
["${aws_subnet.private_subnet-mcp.id}","${aws_subnet.private_subnet2-mcp.id}"]
  min_size             = 2
  max_size             = 2
  target_group_arns     = ["${aws_lb_target_group.alb_target_mcp.arn}"]
  health_check_type     = "EC2"
  tags = [
    {
      key                 = "Name"
      value               = "WebServer"
      propagate_at_launch = true
    },
    {
      key                 = "Name"
      value               = "WebServer"
      propagate_at_launch = true
    }
  ]
}

# * * * * * * * * * DNS * * * * * * * * *
```

```
resource "aws_route53_zone" "primary" {
  name = "mycloudprices.com"
}

resource "aws_route53_record" "www" {
  zone_id = "${aws_route53_zone.primary.zone_id}"
  name    = "www"
  type    = "A"
  alias {
    name                   = "${aws_lb.alb-mcp.dns_name}"
    zone_id                = "${aws_lb.alb-mcp.zone_id}"
    evaluate_target_health = false
  }
}

resource "aws_route53_record" "www2" {
  zone_id = "${aws_route53_zone.primary.zone_id}"
  name    = "mycloudprices.com"
  type    = "A"
  alias {
    name                   = "${aws_route53_record.www.fqdn}"
    zone_id                = "${aws_route53_zone.primary.zone_id}"
    evaluate_target_health = false
  }
}

# * * * * * * * * * Output * * * * * * * * *

# Output
output "dns-alb" {
  value = "${aws_lb.alb-mcp.dns_name}"
  description = "DNS Application Load Balancer"
}

output "endpoint-alb" {
  value = "${aws_docdb_cluster.docdb-mcp.endpoint}"
  description = "Endpoint connection database documentDB"
}

/*
# Template User Data
data "template_file" "init" {
  template = "${file("init.tpl")}"
}
```

```
# Create instance 1
resource "aws_instance" "webserver-mcp" {
  ami               = "${var.ec2_info[0]}"
  instance_type     = "${var.ec2_info[1]}"
  availability_zone = "${var.az[0]}"
  key_name          = "${var.key_name}"
  vpc_security_group_ids = ["${aws_security_group.web-sg-mcp.id}"]
  subnet_id         = "${aws_subnet.public_subnet-mcp.id}"
  user_data         = "${data.template_file.init.rendered}"
  tags {
    Name = "webserver-mcp"
  }
}
*/
```

Anexo B

sVariables.tf

```
variable "access_key" {}
variable "secret_key" {}
variable "region" {
  default = "us-east-1"
}
variable "az" { type = "list" }
variable "cidrs_vpc" { type = "list" }
variable "ec2_info" { type = "list" }
variable "key_name" {}
variable "userdb" {}
variable "passdb" {}
variable "port" {}
variable "databasename" {}
```

Anexo C
Terraform.tfvars

```
access_key = "AKIAJFU3JL7XENKNIZFQ23F"
secret_key = "h6CmRm/5u7gpY5NZRfBoWbpZWkER2iiBleoPljFBF32"
region = "us-east-1"
az = [ "us-east-1e", "us-east-1f" ]
cidrs_vpc = [ "10.0.0.0/16", "10.0.0.0/24", "10.0.1.0/24", "10.0.2.0/23",
"10.0.4.0/23", "10.0.6.0/23" ]
ec2_info = [ "ami-02da3a138888ced85", "t2.micro" ]
key_name = "amazonvm"
userdb = "admindocdb"
passdb = "passworddocdb"
port = "27017"
databasename = "mycloudprices"
```